MP2 系列运动控制 PLC 用户手册（基本指令篇）

编写人	PLC 产品部
手册版本	V0.3
编写日期	2021.1.7

深圳市雷赛智能控制股份有限公司

目录

1 软元件

1.1 MP2 软元件编号一览

进制	点数	范围	说明
输入输出寄存器			
8	1024	X0~X1777	信号输入点
8	1024	Y0~Y1777	信号输出点
辅助继电器			
10	7680	M0~M7679	辅助继电器范围
		M0~M1023	锁存/掉电保持设置范围
		M500~M1023	默认锁存范围
10	512	M8000~M8511	特殊辅助继电器
状态继电器			
10	4096	S0~S4095	状态继电器范围
		S0~S999	锁存/掉电保持设置范围
		S500~S999	默认锁存范围
位元件			
10	32768	B0~B32767	位元件（默认掉电保持）
10	32768	L0~L32767	位元件（默认掉电保持）
10	4000	F0~F7999	位元件（默认掉电保持）
定时器			
10	512	T0~T191	100ms 定时器
		T192~T199	100ms 定时器，子程序用
		T200~T245	10ms 定时器
		T246~T249	1ms 累计定时器
		T250~T255	100ms 累计定时器
		T256~T511	1ms 定时器
计数器			
10	243	C0~C199	16 位计数器
		C200~C234	32 位计数器
		C235~C242	32 位高速计数器
数据寄存器			
10	8000	D0~D7999	数据寄存器范围
		D0~D511	锁存设置范围
		D200~D511	默认锁存范围
		D0~D7999	掉电保持设置范围
10	512	D8000~D8511	特殊寄存器
扩展数据寄存器			
10	16384	R0~R16383	默认断电保持型，轴参数占用

10	16384	RD0~RD16383	默认断电保持型
10	16384	VD0~VD16383	默认保持型
子程序标号			
10	4096	P0~P4095	子程序指针
10	6	I0~I501	I00 □，I10 □，I20□，I30 □，I40 □，I50 □ 为外部中断子程序。 其中□内的数值为 0 时为下降沿中断，为 1 时为上升沿中断。
10	3	I6XX~I8XX	定时中断,指定 1~99ms
10	5	I010~I050	计数器中断，DHSCS 指令用
局部位寄存器			
10	1024	LB0~LB1023	每个功能享有独立的局部变量，但是 Main 和子程序共享同一份变量。
局部字寄存器			
10	1024	LW0~LW1023	每个功能享有独立的局部变量，但是 Main 和子程序共享同一份变量。
局部定时器寄存器			
10	256	LT0~LT255	每个功能享有独立的局部变量，但是 Main 和子程序共享同一份变量。 指令示例：TIM LT0 K21 K100,定时时间=21*100=2100ms
变址寄存器			
10	16	V0~V7 Z0~Z7	通过在指令的操作数中组合使用其他的软元件编号和数值，从而在程序中更改软元件的编号和数值内容的特殊寄存器。  修饰 32 位应用指令中的软元件时，或者处理超出 16 位数值的范围时必须使用 Z0~Z7，入上图所示的 V,Z 组合，由于可编程控制器将 Z 作为 32 位寄存器的低位侧动作，所以即使指定了高位侧的 V0~V7 也不会执行修饰。此外，作为 32 位指定时，会同时参考 V（高位），Z（低位），因此一旦 V 中留有别的用途的数值时，会变成相当大的数值，从而发生运算错误。

特别说明：

掉电保持范围

1.2 MP2 特殊内存配置

特殊继电器一览表

元件地址	软件描述
M8000	Run 时 ON，Stop 时 OFF
M8001	Run 时 OFF，Stop 时 ON
M8002	运行时第一个周期为 ON，其他为 OFF
M8003	运行时第一个周期为 OFF，其他为 ON
M8004	全局错误标记，所有类型错误发生都会置 ON
M8005	电池电压过低，系统无效
M8006	电池电压过低锁存，系统无效
M8007	PLC 退出标记
M8008	用户 LED 映射控制
M8011	10ms 时钟，5msON 5msOFF
M8012	100ms 时钟，50msON 50msOFF
M8013	1s 时钟，500msON 500msOFF
M8014	1min 时钟，30sON 30sOFF
M8015	停止计时以及预置实时时钟用

M8016	时间读出后的显示被停止实时时钟用
M8017	±30 秒的修正实时时钟用
M8018	检测出安装(一直为 ON)实时时钟用
M8019	实时时钟(RTC)错误实时时钟用
M8020	零位标志、加减法运算结果为 0 时接通
M8021	借位标志、减法运算结果超过最大的负值时接通
M8022	进位标志、加法运算结果发生进位时，或者移位结果发生溢出时接通
M8024	指定 BMOV 的方向，ON 时反向传送
M8025	HSC 模式
M8026	RAMP 模式
M8028	
M8029	指令动作结束时接通；
M8031	非保持内存全部清除
M8032	保持内存全部清除
M8033	内存保持停止；ON 时 RUN-STOP 期间不清除非保持区域
M8037	强制 STOP 指令 外部输入强制停止
M8039	M8039 接通后，一直等待到 D8039 中指定的扫描时间到可编程控制器执行这样的循环运算。
M8040	禁止转移
M8046	STL 状态动作，当 M8047 接通时，S0 ～S899、S1000 ～S4095*3 中任意一个为 ON 则接通
M8047	STL 监控有效
M8048	信号报警器动作，当 M8049 接通时，S900～S999 中任意一个为 ON 则接通
M8050	外部中断 I00X 控制
M8051	外部中断 I10X 控制
M8052	外部中断 I20X 控制
M8053	外部中断 I30X 控制
M8054	外部中断 I40X 控制
M8055	外部中断 I50X 控制
M8056	定时器中断 I60X 控制
M8057	定时器中断 I70X 控制
M8058	定时器中断 I80X 控制

M8059	计数器中断 I010~I060 控制
M8061	PLC 硬件出错
M8062	PLC/PP 通信出错
M8063	串行通信出错
M8064	参数出错
M8067	运算出错
M8068	运算出错锁存
M8069	I/O 总线检测
M8076	轴出错，例如同时执行两条 MCTBL 为同一个轴
M8088	在线下载时不检测通讯指令使用
M8089	内部使用 有运动指令或者通讯指令在运行
M8090	BK 类指令完成标记
M8091	字符模式
M8097	单圈调试控制
M8098	单圈调试触发
M8109	输出刷新出错
M8110	不刷新输入
M8111	不刷新输出
M8121	RS 指令 发送待机标志位
M8122	RS 指令 发送请求
M8123	RS 指令 接收结束标志位
M8126	全局信号标志
M8129	RS 指令 判断超时的标志位
M8150	TCP 服务器 1 暂缓发送
M8151	TCP 服务器 2 暂缓发送
M8152	TCP 服务器 3 暂缓发送
M8153	TCP 服务器 4 暂缓发送
M8161	8 位处理模式
M8162	
M8165	SORT2 指令排序方向
M8168	SMOV 指令处理 HEX 数据的功能
M8170	输入 X0 脉冲捕捉

M8171	输入 X1 脉冲捕捉
M8172	输入 X2 脉冲捕捉
M8173	输入 X3 脉冲捕捉
M8174	输入 X4 脉冲捕捉
M8175	输入 X5 脉冲捕捉
M8176	输入 X6 脉冲捕捉
M8177	输入 X7 脉冲捕捉
M8198	备用
M8199	备用
M8200	计数器 C200 计数方向 ON:减计数 OFF:增计数
M8201	计数器 C201 计数方向 ON:减计数 OFF:增计数
M8202	计数器 C202 计数方向 ON:减计数 OFF:增计数
M8203	计数器 C203 计数方向 ON:减计数 OFF:增计数
M8204	计数器 C204 计数方向 ON:减计数 OFF:增计数
M8205	计数器 C205 计数方向 ON:减计数 OFF:增计数
M8206	计数器 C206 计数方向 ON:减计数 OFF:增计数
M8207	计数器 C207 计数方向 ON:减计数 OFF:增计数
M8208	计数器 C208 计数方向 ON:减计数 OFF:增计数
M8209	计数器 C209 计数方向 ON:减计数 OFF:增计数
M8210	计数器 C210 计数方向 ON:减计数 OFF:增计数
M8211	计数器 C211 计数方向 ON:减计数 OFF:增计数
M8212	计数器 C212 计数方向 ON:减计数 OFF:增计数
M8213	计数器 C213 计数方向 ON:减计数 OFF:增计数
M8214	计数器 C214 计数方向 ON:减计数 OFF:增计数
M8215	计数器 C215 计数方向 ON:减计数 OFF:增计数
M8216	计数器 C216 计数方向 ON:减计数 OFF:增计数
M8217	计数器 C217 计数方向 ON:减计数 OFF:增计数
M8218	计数器 C218 计数方向 ON:减计数 OFF:增计数
M8219	计数器 C219 计数方向 ON:减计数 OFF:增计数
M8220	计数器 C220 计数方向 ON:减计数 OFF:增计数
M8221	计数器 C221 计数方向 ON:减计数 OFF:增计数
M8222	计数器 C222 计数方向 ON:减计数 OFF:增计数

M8223	计数器 C223 计数方向 ON:减计数 OFF:增计数
M8224	计数器 C224 计数方向 ON:减计数 OFF:增计数
M8225	计数器 C225 计数方向 ON:减计数 OFF:增计数
M8226	计数器 C226 计数方向 ON:减计数 OFF:增计数
M8227	计数器 C227 计数方向 ON:减计数 OFF:增计数
M8228	计数器 C228 计数方向 ON:减计数 OFF:增计数
M8229	计数器 C229 计数方向 ON:减计数 OFF:增计数
M8230	计数器 C230 计数方向 ON:减计数 OFF:增计数
M8231	计数器 C231 计数方向 ON:减计数 OFF:增计数
M8232	计数器 C232 计数方向 ON:减计数 OFF:增计数
M8233	计数器 C233 计数方向 ON:减计数 OFF:增计数
M8234	计数器 C234 计数方向 ON:减计数 OFF:增计数
M8260	MODBUS 指令完成标记
M8261	MODBUS 指令错误标记
M8304	乘法除法指令结果为 0 标记
M8305	除法指令溢出
M8328	BFM 指令暂停
M8329	通讯指令发生错误
M8330	DUTY 指令使用
M8331	DUTY 指令使用
M8332	DUTY 指令使用
M8333	DUTY 指令使用
M8334	DUTY 指令使用
M8398	1ms 的环形计数（32 位动作）
M8401	
M8402	
M8403	
M8409	
M8421	
M8422	
M8423	
M8429	

M8438	
M8500	
M8501	

特殊寄存器一览表

元件地址	软件描述
D8000	看门狗定时器
D8003	驱动库类型 1=mcsim 20=mcpci 30=mccat
D8010	扫描当前值
D8011	扫描时间的最小值
D8012	扫描时间的最大值
D8013	0～59 秒(实时时钟用)
D8014	0～59 分(实时时钟用)
D8015	0～23 小时(实时时钟用)
D8016	1～31 日(实时时钟用)
D8017	1～12 月(实时时钟用)
D8018	西历 2 位数(0～99)(实时时钟用)
D8019	0(日)～6(六)(实时时钟用)
D8020	输入滤波时间
D8039	恒定扫描时间
D8061	硬件出错的错误代码编号
D8062	通信出错的错误代码编号
D8063	串行通信出错（COM1）的错误代码编号
D8064	参数出错的错误代码编号
D8067	运算出错的错误代码编号
D8068	发生运算出多的步编号的锁存
D8069	M8065~7 的产生出错的步编号
D8070	出错的功能块名称
D8076	轴错误代码
D8077	软件版本
D8082	软件版本(低位)
D8083	软件版本(高位)
D8084	硬件错误 1
D8085	硬件错误 2

D8108	特殊模块的连接台数
D8120	RS 指令设定通信格式
D8122	RS 指令 发送数据的剩余点数
D8123	RS 指令 接收点数的监控
D8124	RS 指令 报头<初始值：STX>
D8125	RS 指令 报尾<初始值：ETX>
D8129	RS 指令的设定超时时间
D8150	TCP 服务器 1 发送计数
D8151	TCP 服务器 2 发送计数
D8152	TCP 服务器 3 发送计数
D8153	TCP 服务器 4 发送计数
D8164	
D8201	扩展模块 01 类型
D8202	扩展模块 02 类型
D8203	扩展模块 03 类型
D8204	扩展模块 04 类型
D8205	扩展模块 05 类型
D8206	扩展模块 06 类型
D8207	扩展模块 07 类型
D8208	扩展模块 08 类型
D8209	扩展模块 09 类型
D8210	扩展模块 10 类型
D8211	扩展模块 11 类型
D8212	扩展模块 12 类型
D8213	扩展模块 13 类型
D8214	扩展模块 14 类型
D8215	扩展模块 15 类型
D8216	扩展模块 16 类型
D8217	扩展模块 17 类型
D8218	扩展模块 18 类型
D8219	扩展模块 19 类型
D8220	扩展模块 20 类型
D8235	高速计数器 C235 模式设置

D8236	高速计数器 C236 模式设置
D8237	高速计数器 C237 模式设置
D8238	高速计数器 C238 模式设置
D8239	高速计数器 C239 模式设置
D8240	高速计数器 C240 模式设置
D8241	高速计数器 C241 模式设置
D8242	高速计数器 C242 模式设置
D8280	状态机当前指令
D8310	RND 生成随机数用的数据
D8330	DUTY 指令 定时时钟输出 1
D8331	DUTY 指令 定时时钟输出 2
D8332	DUTY 指令 定时时钟输出 3
D8333	DUTY 指令 定时时钟输出 4
D8334	DUTY 指令 定时时钟输出 5
D8396	记录软件运行的时间(10ms 单位)
D8400	RS2 端口 1 通讯设置
D8402	RS2 端口 1 发送剩余数量
D8403	RS2 端口 1 接收数量
D8409	RS2 端口 1 接收超时时间
D8410	RS2 端口 1STX 字符 1
D8411	RS2 端口 1STX 字符 2
D8412	RS2 端口 1EXT 字符 1
D8413	RS2 端口 1EXT 字符 2
D8420	RS2 端口 2 通讯设置
D8422	RS2 端口 2 发送剩余数量
D8423	RS2 端口 2 接收数量
D8429	RS2 端口 2 接收超时时间
D8430	RS2 端口 2STX 字符 1
D8431	RS2 端口 2STX 字符 2
D8432	RS2 端口 2EXT 字符 1
D8433	RS2 端口 2EXT 字符 2
D8438	RS2 指令错误代码
D8492	最大支持轴数

D8493	MAC 地址 5-6
D8494	MAC 地址 3-4
D8495	MAC 地址 1-2
D8496	可视化用户切换
D8497	可视化当前用户
D8498	可视化当前页面
D8499	可视化页面切换
D8500	IP 地址 1
D8501	IP 地址 2
D8502	IP 地址 3
D8503	IP 地址 4
D8507	轴数量
D8508	禁止轴信息刷新 00~15
D8509	禁止轴信息刷新 16~31

2 编程软件

2.1 编程软件环境要求

mPLC 系列采用 mPLC Studio 编程软件进行编程，mPLC Studio 安装需要最低配置如下。
操作系统:Windows XP，Windows 7，Windows 8，Windows 10
内存:1G 以上
硬盘:空闲 1G 以上
CPU: intel i3 以上或 AMD 同等级 CPU

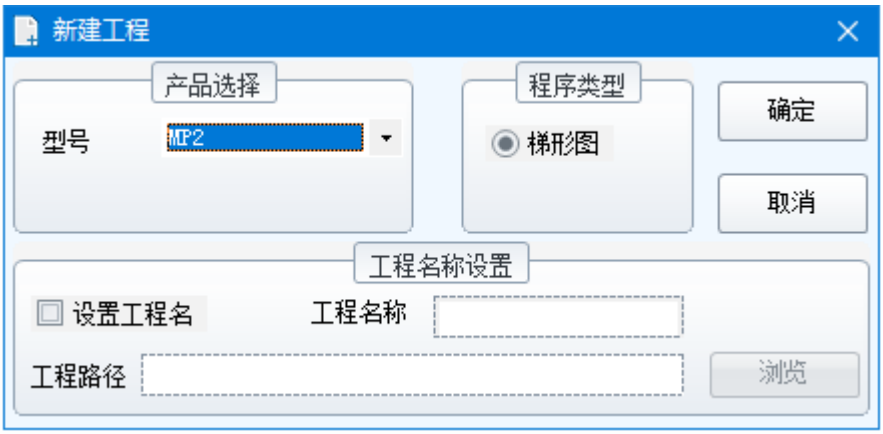
2.2 通讯电缆

1. RS232-Mini DIN8(PORT0)
2. RS485(PORT2)
3. 以太网(RJ45)

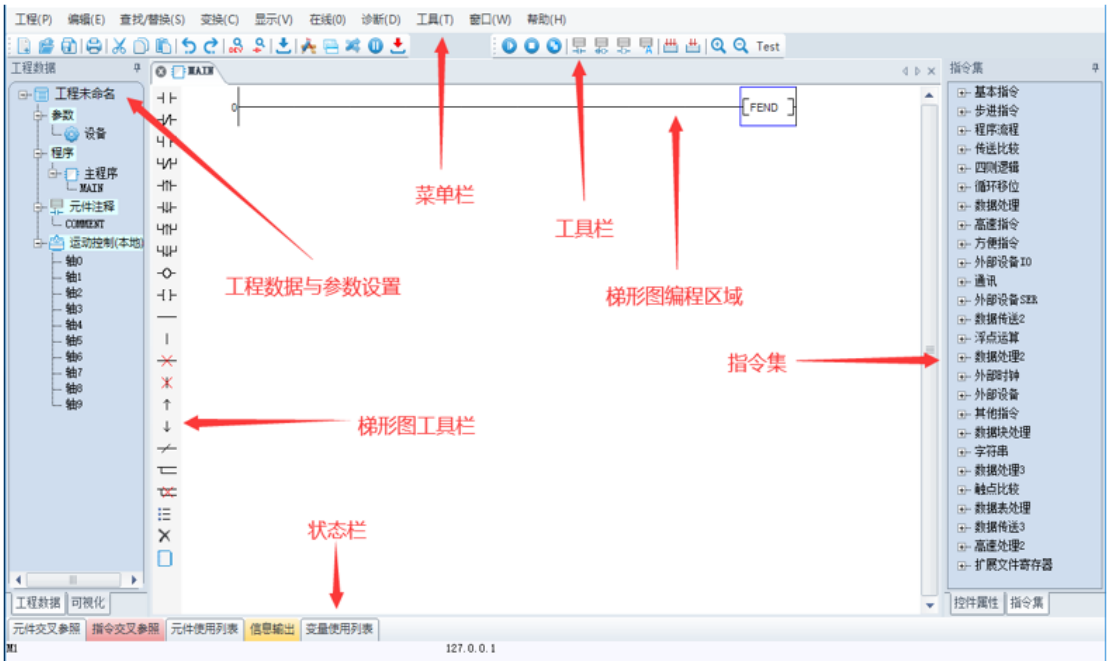
可用作以上的通讯电缆与 mPLC 进行通讯，对于没有 DB9 型的 RS232 串口的电脑，可用 USB 转 RS232 进行下载以及调试。

2.3 mPLC Studio 软件简介

点击左上角【工程】-【新建工程】或按快捷键 CTRL+N 后弹出以下窗口。



填写上相关信息后点击【确定】按钮，即可新建一个新的工程。



2.4 程序输入

1. 在梯形图工具栏中，点击指令，将其添加到梯形图中。
2. 在右侧的【指令集】中选择指令。
3. 如熟悉应用指令，可直接通过键盘手动输入指令。

2.5 工程编译

1. 可通过快捷键“F4”进行编译转换程序  图标左边的为编译所有工程，图标右边的为编译当前工程。
2. 可通过菜单中的【变换】-【变换】来进行编译转换程序。
3. 如果工程在编译过程中没有出现错误提示，那么此工程将可以进行下载。

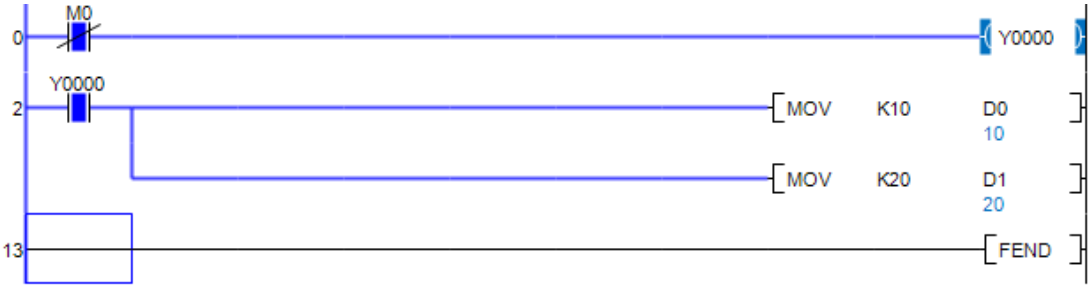
2.6 工程下载/上载方式

1. 使用编程电缆把PLC 与PC 连接。
2. 设置相应的通信方式。菜单选择【在线】-【通信设置】。

3. 菜单选择【在线】-【读取】/【写入】。或者点击工具栏上的  来进行上传或下载。

2.7 监控以及调试

1. 当程序下载后，点击菜单栏上的  左边的是开始监控，中间的是停止监控，右边的是停止全部监控，当点击开始监控后，编程软件进入在线监控模式。



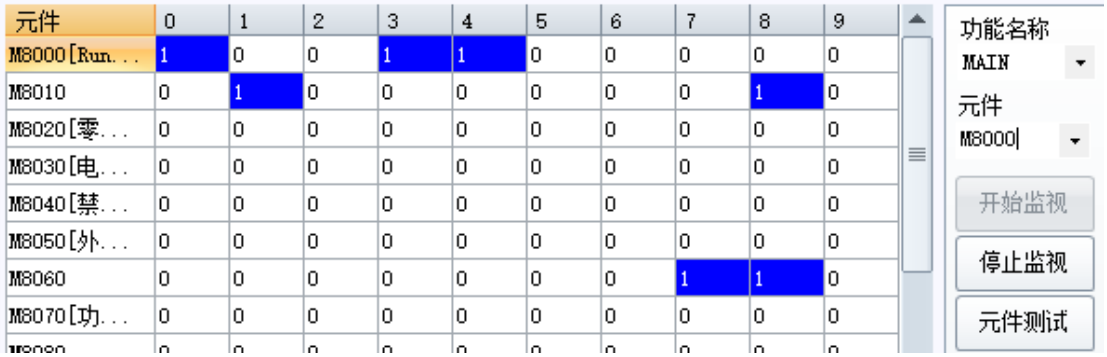
2. 当程序中显示如下所示的状态，说明已经进入了在线监控状态。

3. 进入在线监控状态后，可点击图标  对软元件进行赋值修改。

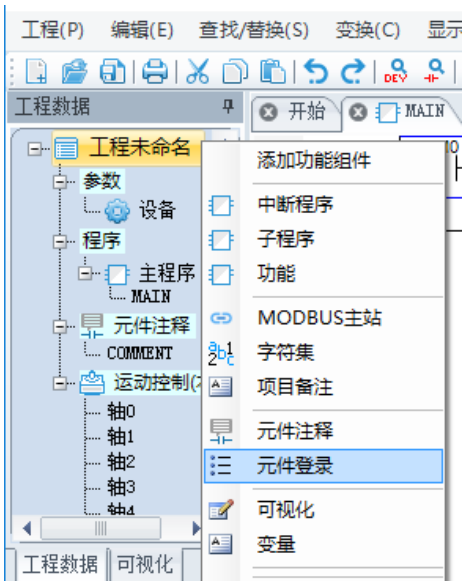
4. 出现此界面后，点击梯形图中的软元件即可跳转至软元件进行 ON、OFF 等操作。

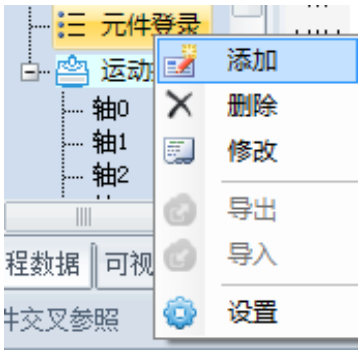
2.8 元件监控表

点击 图标可弹出数据监控表，在右侧点击元件，输入想要监控的元件地址，可显示出需要监控的地址一直至此元件结束地址，例如：



如不想查看所有，可单独添加元件登录



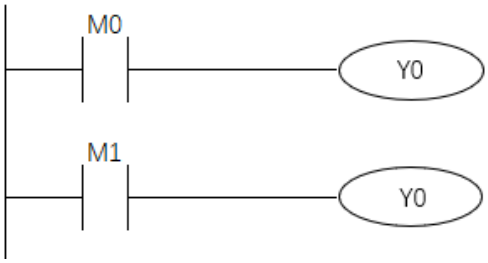


在元件登陆中添加元件登录表，添加软元件地址，点击开始监控。

元件	当前值	数据类型	显示方式
D0	10	16位整数	DEC
D1	20	16位整数	DEC

2.9 双线圈

双线圈并没有违背程序的编辑原则，但结果可能不是用户所希望的输出状态。因为PLC的实际端口I/O 都是在程序结束时刷新。所以仅仅程序的最后一个状态会被刷新。中间的变化状态并不会表现出来。



程序分析:

当 M0 为 ON M1 为 OFF 时，在 PLC 一个扫描周期内，会有两种不同的状态，从而导致一个扫描周期内状态翻转两次。

同时以最后一个的状态位为准，在 PLC 的扫描过程中，刷新 IO 是在每一次扫描周期结束后刷新当前的 IO 状态。

2.10 PLC 的启动与停止

点击编程软件的菜单栏上的【在线】-【远程操作】可对其进行RUN/OFF的操作，菜单栏中也有快捷按钮可以操作

2.11 查找软元件

1. 使用快捷键【Ctrl+F】进行搜索软元件以及程序。
2. 或点击菜单栏中的【查找/替换】-【元件查找】进行搜索软元件以及程序。

2.12 单圈调试

单圈调试指不通过扫描周期循环扫描，而是只运算一圈，此模式需要配合特殊继电器来使用，将【M8097】（单圈调试控制）使能，同时将【M8098】（单圈调试触发）置ON一次程序就扫描一次。

2.13 在线修改

当程序在监控中时，可以直接修改程序后再进行编译 后可点击在线下载按钮 即可将程序下载至RAM中（断电非保持，如需要保持，请重新下载一遍程序至Flash内）。

3 指令说明

MP2 系列支持使用的指令集如下表。

基本指令	
指令	指令说明
LD	装载
LDI	装载取反
LDP	取脉冲上升沿

LDF	取脉冲下降沿
AND	与
ANI	与非
ANDP	与脉冲上升沿
ANDF	与脉冲下降沿
OR	或
ORI	或非
ORP	或脉冲上升沿
ORF	或脉冲下降沿
MEP	运算结果上升沿脉冲化指令
MEF	运算结果下降沿脉冲化指令
ANB	回路块与
ORB	回路块或
MPS	压入堆栈
MRD	读取堆栈
MPP	弹出堆栈
INV	取反
OUT	输出
SET	置位
RST	复位
PLS	上升沿微分输出
PLF	下降沿微分输出
MC	主控
MCR	主控复位
NOP	空操作
END	程序结束

STL	步进梯形图开始
RET	步进梯形图返回
程序流程指令	
指令	说明
CJ	条件跳转
CALL	子程序调用
SRET	子程序返回
IRET	中断返回
EI	允许中断
DI	禁止中断
FEND	主程序结束
WDT	看门狗定时器
FOR	循环范围的开始
NEXT	循环范围的结束
CALLFC	调用功能程序
FCEND	功能结束
FCRET	功能退出
传送比较指令	
指令	说明
CMP	比较
ZCP	区间比较
MOV	传送
SMOV	移位
CML	取反传送
BMOV	成批传送
FMOV	多点传送

XCH	交换
BCD	BCD 转换
BIN	BIN 转换
四则逻辑运算指令	
指令	说明
ADD	BIN 加法
SUB	BIN 减法
MUL	BIN 乘法
DIV	BIN 除法
INC	BIN 加 1
DEC	BIN 减 1
WAND	逻辑与
WOR	逻辑或
WXOR	逻辑异或
NEG	求补码
循环移位指令	
指令	说明
ROR	循环右移
ROL	循环左移
RCR	带进位循环右移
RCL	带进位循环左移
SFTR	位右移
SFTL	位左移
WSFR	字右移
WSFL	字左移
SFWR	移位写入

SFRD	移位读出
SFWR2	移位写入
SFRD2	移位读出
数据处理指令	
指令	说明
ZRST	成批复位
ZSET	成批置位
DECO	译码
ENCO	编码
SUM	ON 位总数
BON	ON 的判定
MEAN	平均值
ANS	信号报警器置位
ANR	信号报警器复位
SQR	BIN 开方
FLT	BIN 整数->2 进制浮点数转换
高速处理指令	
指令	说明
HSCS	比较置位（高速计数器）
HSCR	比较复位（高速计数器）
HSZ	区间比较（高速计数器）
定位指令	
指令	说明
MCTBL	轴定位控制
MCZERO	轴归零操作
方便指令	

指令	说明
SER	数据检索
ABSD	凸轮控制绝对方式
INCD	凸轮控制相对方式
TTMR	示教定时器
STMR	特殊定时器
ALT	交替输出
RAMP	斜坡信号
SORT	数据排序
外部设备 IO	
指令	说明
SEGD	七段码译码
ASC	ASCII 数据输入
FROM	BFM 的读出
TO	BFM 的写入
通讯指令	
指令	说明
MODBUS	MODBUS 主站指令
OPENS	通讯端口控制
SENDS	发送字符串
REVS	接收字符串
外部设备 SER	
指令	说明
PRUN	八进制位传送
ASCI	HEX 到 ASCII 的转换
HEX	ASCII 到 HEX 的转换

CCD	校验码
VRRD	
数据传送 2	
ZPUSH	变址寄存器的成批保存
ZPOP	变址寄存器的恢复
浮点运算指令	
指令	说明
ECMP	2 进制浮点数比较
EZCP	2 进制浮点数区间比较
EMOV	2 进制浮点数数据传送
EBCD	2 进制浮点数→10 进制浮点数转换
EBIN	10 进制浮点数→2 进制浮点数转换
EADD	2 进制浮点数加法运算
ESUB	2 进制浮点数减法运算
EMUL	2 进制浮点数乘法运算
EDIV	2 进制浮点数除法运算
EXP	2 进制浮点数指数运算
LOGE	2 进制浮点数自然对数运算
LOG10	2 进制浮点数常用对数运算
ESQR	2 进制浮点数开方运算
ENEG	2 进制浮点数符号翻转
INT	2 进制浮点数→BIN 整数的转换
SIN	2 进制浮点数 SIN 运算
COS	2 进制浮点数 COS 运算
TAN	2 进制浮点数 TAN 运算
ASIN	2 进制浮点数 SIN-1 运算

ACOS	2 进制浮点数 COS-1 运算
ATAN	2 进制浮点数 TAN-1 运算
RAD	2 进制浮点数角度→弧度的转换
DEG	2 进制浮点数弧度→角度的转换
数据处理 2 指令	
WSUM	算出数据合计值
WTOB	字节单位的数据分离
BTOW	字节单位的数据结合
UNI	16 位数据的 4 位结合
DIS	16 位数据的 4 位分离
SWAP	高低字节互换
SORT2	数据排序 2
时钟运算	
指令	说明
TCMP	时钟数据比较
TZCP	时钟数据区间比较
TADD	时钟数据加法运算
TSUB	时钟数据减法运算
HTOS	时、分、秒数据的秒转换
STOH	秒数据的[时、分、秒]转换
TRD	时钟数据的读出
TWR	时钟数据的读入
HOUR	长时间计时
TIM	定时器
TIS	周期定时器
格雷码	

指令	说明
GRY	格雷码的转换
GBIN	格雷码的逆转换
其他指令	
COMRD	读出软元件注释数据
RND	产生随机数
DUTY	产生定时脉冲
CRC	CRC 校验
数据块指令	
指令	说明
BK+	数据块的加法运算
BK-	数据块的减法运算
BKCMP=	数据块比较
BKCMP>	
BKCMP<	
BKCMP<>	
BKCMP<=	
BKCMP>=	
字符串控制指令	
指令	说明
$+	字符串的结合
LEN	检测出字符串的长度
RIGHT	从字符串的右侧开始取出
LEFT	从字符串的左侧开始取出
MIDR	从字符串中的任意取出
MIDW	字符串中的任意替换

INSTR	字符串的检索
$MOV	字符串的传送
数据处理指令 3	
指令	说明
FDEL	数据表的数据删除
FINS	数据表的数据插入
POP	读取后入的数据[先入后出控制用]
SFR	16 位数据 n 位右移(带进位)
SFL	16 位数据 n 位左移(带进位)
FDEL2	数据表的数据删除 2
FINS2	数据表的数据插入 2
POP2	读取后入的数据[先入后出控制用]
触点比较指令	
指令	说明
LD=	(S1)=(S2)
LD>	(S1)>(S2)
LD<	(S1)<(S2)
LD<>	(S1)≠(S2)
LD<=	(S1)<=(S2)
LD>=	(S1)>=(S2)
AND=	(S1)=(S2)
AND>	(S1)>(S2)
AND<	(S1)<(S2)
AND<>	(S1)≠(S2)
AND<=	(S1)<=(S2)
AND>=	(S1)>=(S2)

OR=	(S1)=(S2)
OR>	(S1)>(S2)
OR<	(S1)<(S2)
OR<>	(S1)≠(S2)
OR<=	(S1)<=(S2)
OR>=	(S1)>=(S2)
LDE=	(S1)=(S2)浮点数比较
LDE>	(S1)>(S2)浮点数比较
LDE<	(S1)<(S2)浮点数比较
LDE<>	(S1)≠(S2)浮点数比较
LDE<=	(S1)<=(S2)浮点数比较
LDE>=	(S1)>=(S2)浮点数比较
ANE=	(S1)=(S2)浮点数比较
ANE>	(S1)>(S2)浮点数比较
ANE<	(S1)<(S2)浮点数比较
ANE<>	(S1)≠(S2)浮点数比较
ANE<=	(S1)<=(S2)浮点数比较
ANE>=	(S1)>=(S2)浮点数比较
ORE=	(S1)=(S2)浮点数比较
ORE>	(S1)>(S2)浮点数比较
ORE<	(S1)<(S2)浮点数比较
ORE<>	(S1)≠(S2)浮点数比较
ORE<=	(S1)<=(S2)浮点数比较
ORE>=	(S1)>=(S2)浮点数比较
数据表处理	
指令	说明

LIMIT	上下限限位控制
BAND	死区控制
ZONE	区域控制
SCL	定坐标(不同点坐标数据)
DABIN	10 进制 ASCII→BIN 的转换
BINDA	BIN→10 进制 ASCII 的转换
SCL2	定坐标 2(X/Y 坐标数据)
数据传送 3	
指令	说明
RBFM	BFM 分割读出
WBFM	BFM 分割写入
扩展文件寄存器	
指令	说明
LOADR	读出扩展文件寄存器
SAVER	成批写入扩展文件寄存器

注意：指令后面+P 代表边沿触发，指令前面+D 代表 32 位指令。

3.1 基本指令

3.1.1 基本指令大致分类

基本指令大致分五类：

1、触点指令：一般放在梯形图中紧接母线的地方，根据触发方式的不同来作为后面指令执行的一个动作开关。

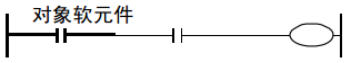
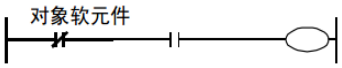
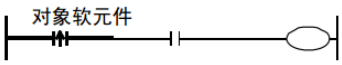
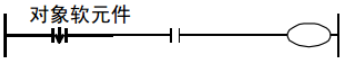
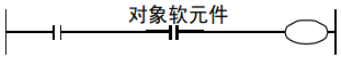
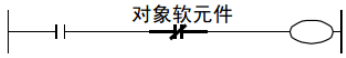
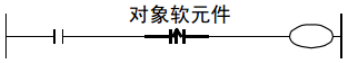
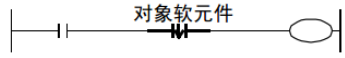
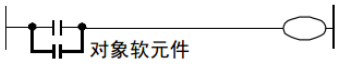
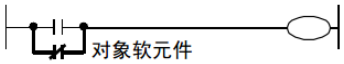
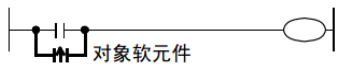
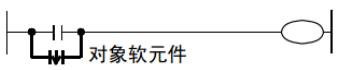
2、结合指令：连接梯形图形成多个执行逻辑。

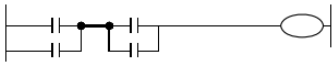
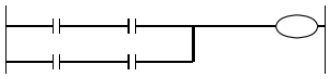
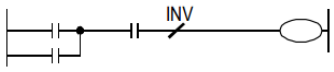
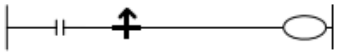
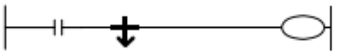
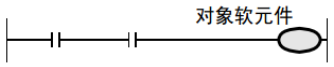
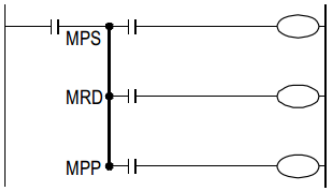
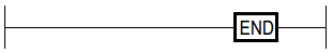
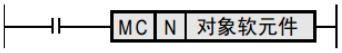
3、输出指令：根据输出方式给对象软元件赋初值。

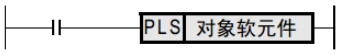
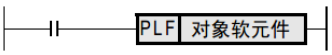
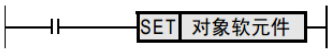
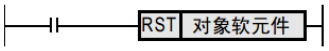
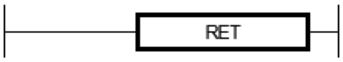
4、主控指令：控制一个程序块的开启关闭。

5、结束指令：END，放于梯形图末尾；FEND 用于主程序结束，会执行与 END 指令相同的输出处理、输入处理、看门狗定时器的刷新，然后返回到 0 步的程序，在编写子程序和中断程序时需要使用 FEND 指令。

基本指令表如下（a 触点：常开触点；b 触点：常闭触点）：

记号	称呼	符号	功能	对象软元件
LD	取		a 触点的逻辑运算开始	X, Y, M, S, T, C, L, B, F, Dx. y
LDI	取反		b 触点的逻辑运算开始	X, Y, M, S, T, C, L, B, F, Dx. y
LDP	取脉冲上升沿		检测到上升沿运算开始	X, Y, M, S, T, C, L, B, F, Dx. y
LDF	取脉冲下降沿		检测到下降沿运算开始	X, Y, M, S, T, C, L, B, F, Dx. y
AND	与		串联 a 触点	X, Y, M, S, T, C, L, B, F, Dx. y
ANI	与反转		串联 b 触点	X, Y, M, S, T, C, L, B, F, Dx. y
ANDP	与脉冲上升沿		上升沿检出的串联连接	X, Y, M, S, T, C, L, B, F, Dx. y
ANDF	与脉冲下降沿		下降沿检出的串联连接	X, Y, M, S, T, C, L, B, F, Dx. y
OR	或		并联 a 触点	X, Y, M, S, T, C, L, B, F, Dx. y
ORI	或反转		并联 b 触点	X, Y, M, S, T, C, L, B, F, Dx. y
ORP	或脉冲上升沿		上升沿检出的并联连接	X, Y, M, S, T, C, L, B, F, Dx. y
ORF	或脉冲下降沿		下降沿检出的并联连接	X, Y, M, S, T, C, L, B, F, Dx. y

ANB	回路块与		回路块的串联连接	-
ORB	回路块或		回路块的并联连接	-
INV	反转		运算结果的反转	-
MEP	运算结果上升沿脉冲化指令		运算结果的上升沿时为 ON	-
MEF	运算结果下降沿脉冲化指令		运算结果的下降沿时为 ON	-
OUT	输出		线圈驱动	Y, M, S, T, C, L, B, F, Dx. y
NOP	空操作		无处理	-
MPS	存储进栈		压入堆栈	-
MRD	存储读栈		读取堆栈	-
MPP	存储出栈		弹出堆栈	-
END	结束		程序结束以及 输入输出处理 和返回 0 步	-
MC	主控		连接到公共触点	Y, M

MCR	主控复位		解除连接到公共触点	–
PLS	脉冲		上升沿微分输出	Y, M, L, B, F
PLF	下降沿脉冲		下降沿微分输出	Y, M, L, B, F
SET	置位		动作保持	Y, M, S, L, B, F, Dx.y
RST	复位		解除保持的动作，清除当前值及寄存器	Y, M, S, L, B, F, T, C, , Dx. y, D, R, RD, VD, V, Z
STL	步进梯形图开始		步进梯形图开始	S
RET	步进梯形图返回		步进梯形图返回	–

3.1.2 基本指令使用中需要注意的地方

1、OUT 指令的使用说明

（1）使用定时器和计数器时

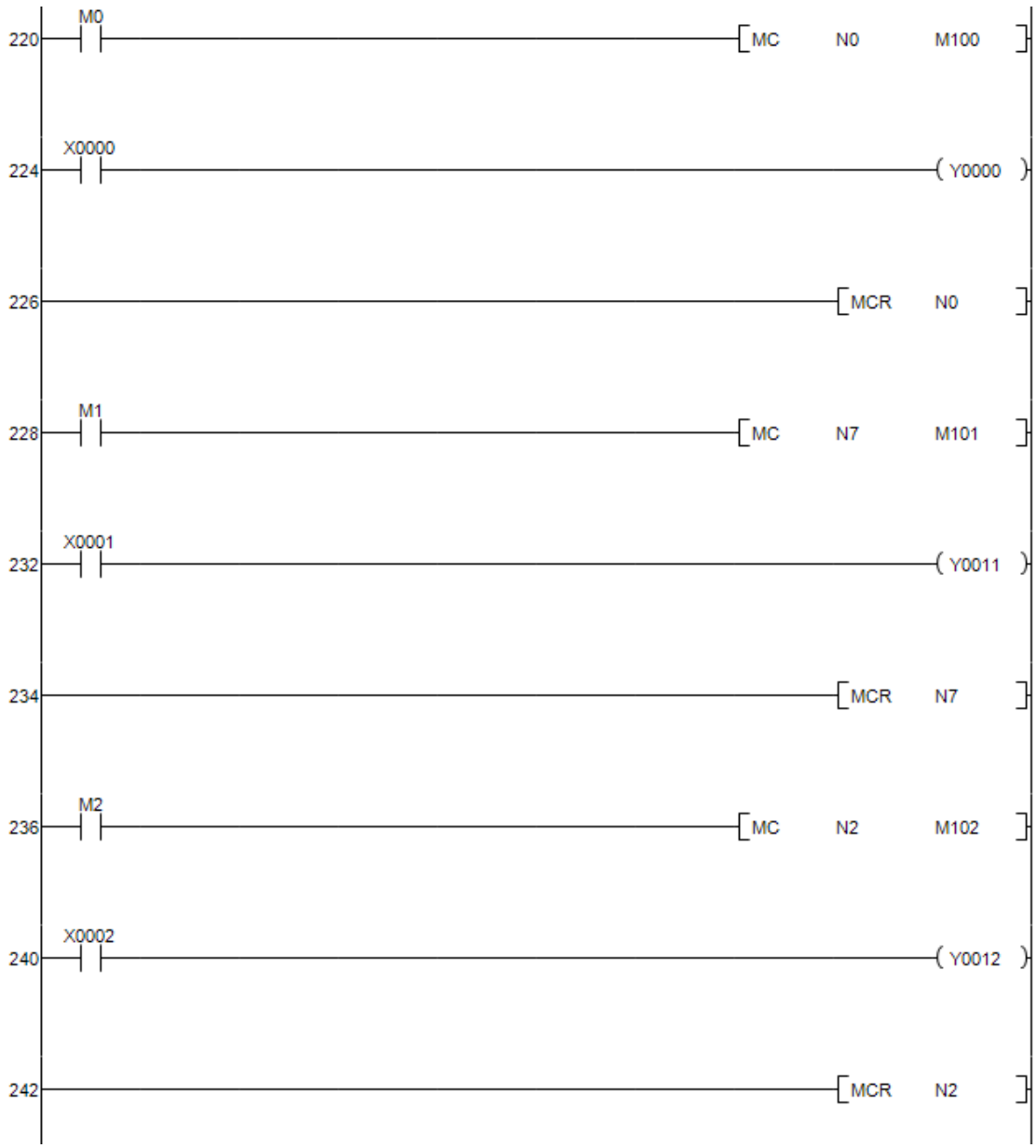
在针对定时器的计时线圈和计数器的计数线圈的 OUT 指令后需要加上设定值。设定值可以使用 10 进制数(K)直接指定，也可以使用数据寄存器(D)或扩展寄存器(R)间接指定。

（2）定时器、计数器的设定范围

定时器和计数器的设定值的设定范围以及实际的定时器常数、OUT 指令的程序步数(包含设定值)参照 1.1 MP2 软元件编号一览。

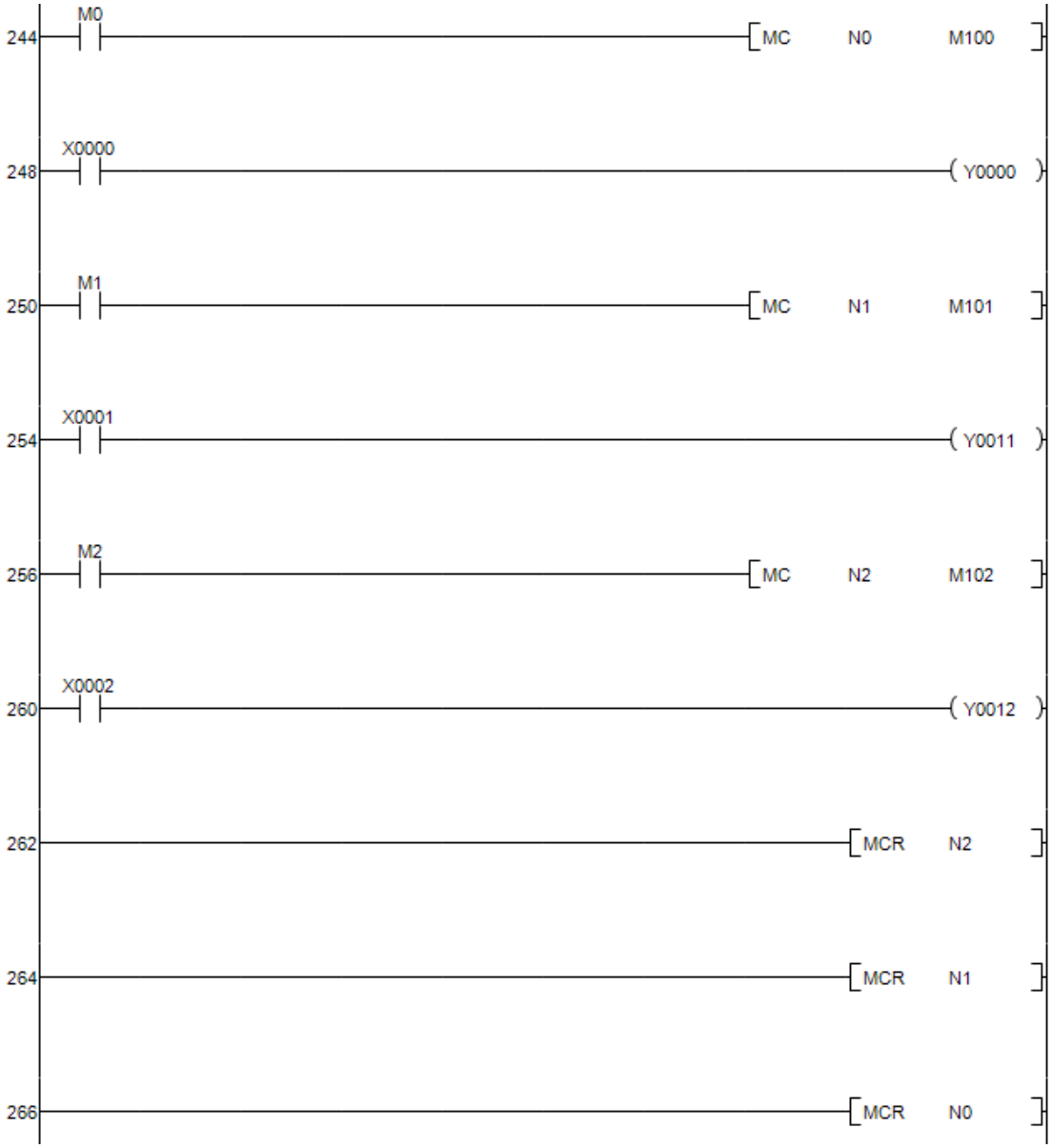
2、主控指令 MC、MCR 使用说明

（1）主控的无嵌套使用



无嵌套时 N 的编号可以任意排列，且使用次数无限制。

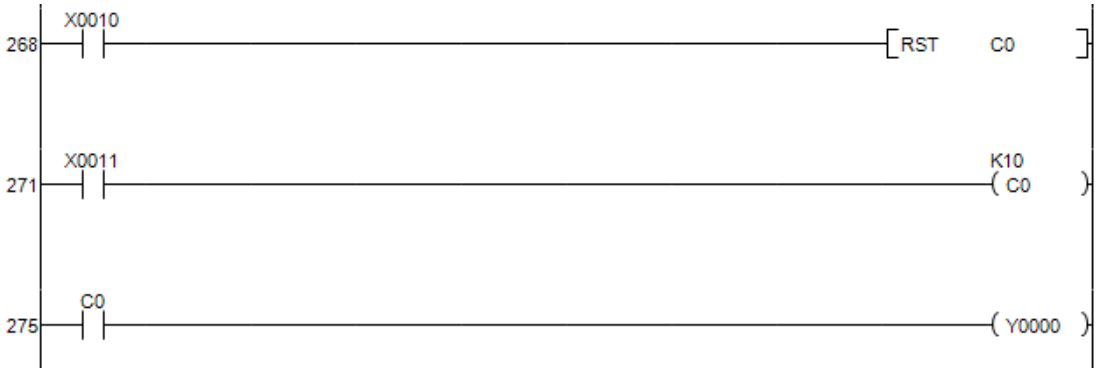
（2）主控的嵌套使用



有嵌套时最大允许 8 层嵌套，执行 MC 指令时，嵌套等级 N 编号依次增大，N0->N7，返回时，执行 MCR 指令，从大的嵌套等级开始解除，N7->N0。

3、RST 复位指令的使用说明

（1）复位普通定时器和累计型计数器



由 C0 对 X011 的 OFF→ON 的次数增计数，计数结果达到设定值 K10 的时候，输出触点 C0 动作。此后，即使 X011 从 OFF 变为 ON，计数器的当前值也不改变，输出触点也保持动作。为了清除这些，恢复输出触点，要使 X010 为 ON。OUT C 指令后面，需要指定常数 K 或间接设定用的数据寄存器编号。停电保持(保持)用计数器的场合，即使停电也能保持当前值以及输出触点的动作状态和复位状态。

（2）复位高速计数器

高速计数器的复位一般不由 RST 控制，高速有增减计数，当增计数达到设定值时置位高速计数器，减计数到 0 时则复位高速计数器。

4、结束指令 END 的使用说明

（1）程序结束处理

可编程控制器重复执行输入处理->执行程序->输出处理，若在程序的最后写入 END 指令，则不执行此后剩余的程序步，而直接进行输出处理，FEND 同 END。

（2）重要参数刷新

执行 END 时会刷新看门狗时间，刷新报错寄存器。

3.2 程序流程

3.2.1 CJ 指令(条件跳转)

1、指令格式

16bit 3 步		32bit		指令格式
CJ	CJP	\	\	CJ Pn

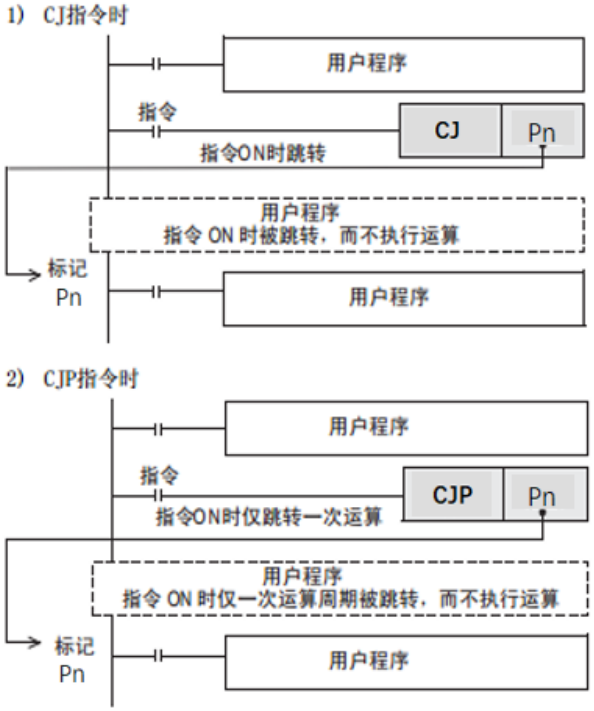
操作数的数据类型：

操作数	内容	类型
Pn	跳转目标标记的指针编号(P)	指针编号

操作数为指针标号 P0~P4095，其中 P63 为 END 所在步序，不需标记。指针标号允许用变址寄存器修改。CJ 和 CJP 都占 3 个程序步，指针标号占 1 步。

2、功能和动作说明

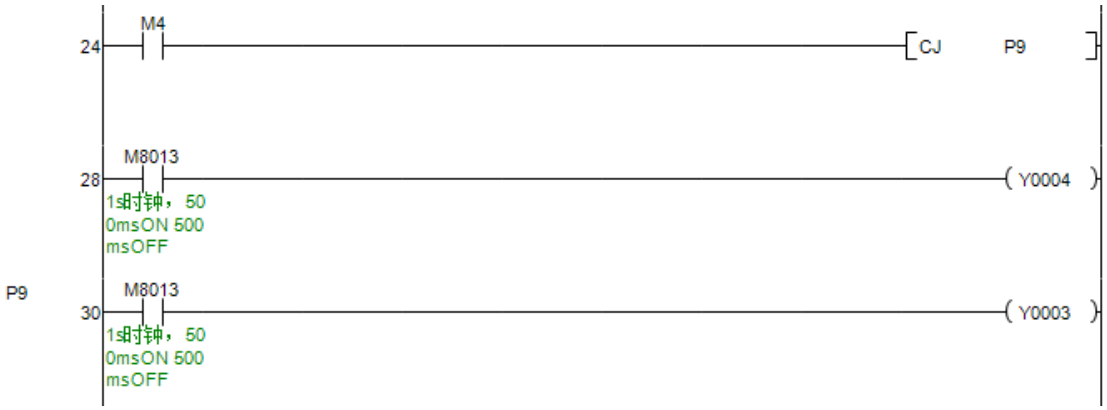
当指令输入为 ON 时，执行指定标记(指针编号)的程序。



3、程序举例

条件跳转指令 CJ 的步骤：

如图所示，当 M4 接通时，则由 CJ　P9 指令跳到标号为 P9 的指令处开始执行，跳过了 Y4 输出部分，只有 Y3 闪烁，减少了扫描周期。如果 M4 断开，跳转不会执行，则程序按原顺序执行，Y3、Y4 呈现同步闪烁。



4、注意事项

（1）在一个程序中一个标号只能出现一次，否则将出错；

（2）在跳转执行期间，即使被跳过程序的驱动条件改变，但其线圈（或结果）仍保持跳转前的状态，因为跳转期间根本没有执行这段程序。

（3）如果在跳转开始时定时器和计数器已在工作，则在跳转执行期间它们将停止工作，到跳转条件不满足后又继续工作。但对于正在工作的高速计数器 C235~C242 或者子程序专用定时器不管有无跳转仍连续工作。

（4）若积算定时器和计数器的复位（RST）指令在跳转区外，即使它们的线圈被跳转，但对它们的复位仍然有效。

（5）跳转指令可以根据特定的需求减少程序周期，跳过不需要运行的程序段。

3.2.2 CALL 指令(子程序调用)

1、指令格式

16bit 3 步		32bit		指令格式
CALL	CALLP	\	\	CALL D

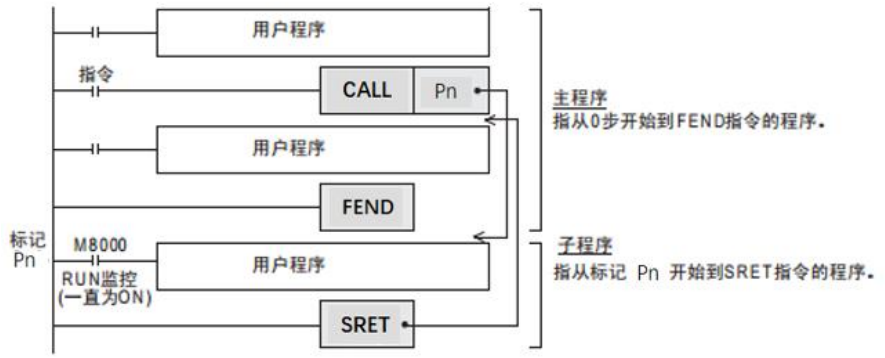
操作数的数据类型如下表

操作数	内容	类型
Pn	跳转目标标记的指针编号(P)	指针编号

操作数为指针标号 P0~P4095

2、功能和动作说明

当指令输入为 ON 时，执行 CALL 指令，向标记 Pn 的步跳转。接着，执行标记 Pn 的子程序。执行 SERT 指令后，返回到 CALL 指令的下一步。

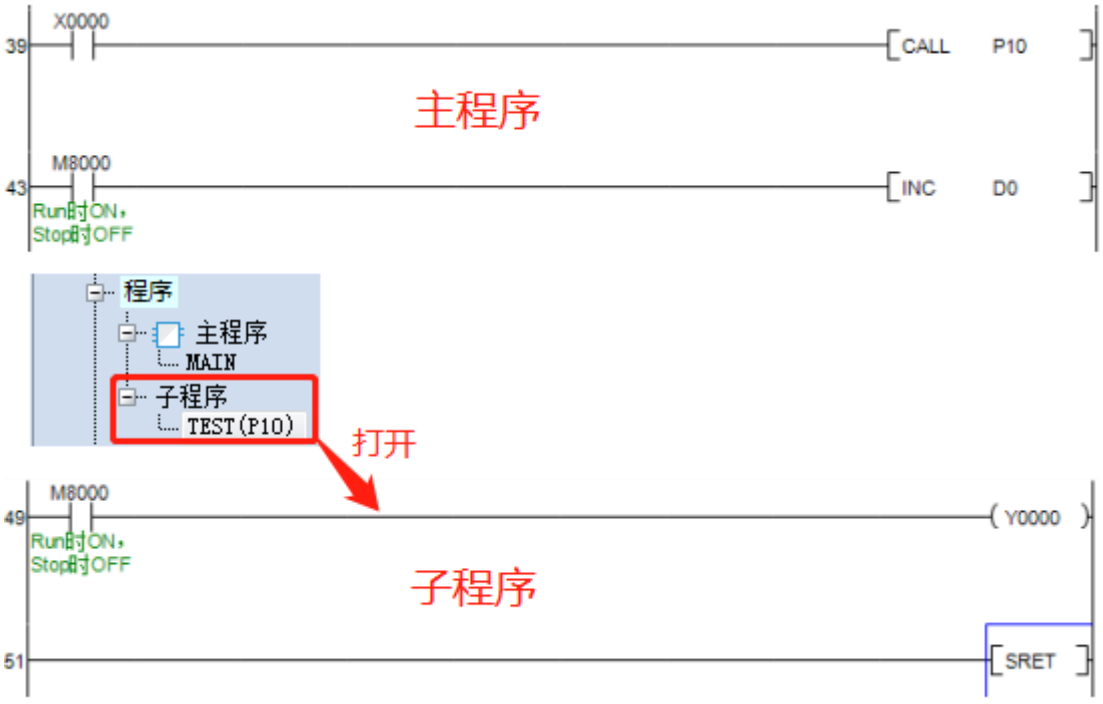


3、程序举例

（1）CALL 无嵌套的使用

主程序：X0 为 ON，则向标记 P10 的步跳转

子程序：执行子程序后，通过执行 SRET 指令，可以返回到原来的步+1 的位置。

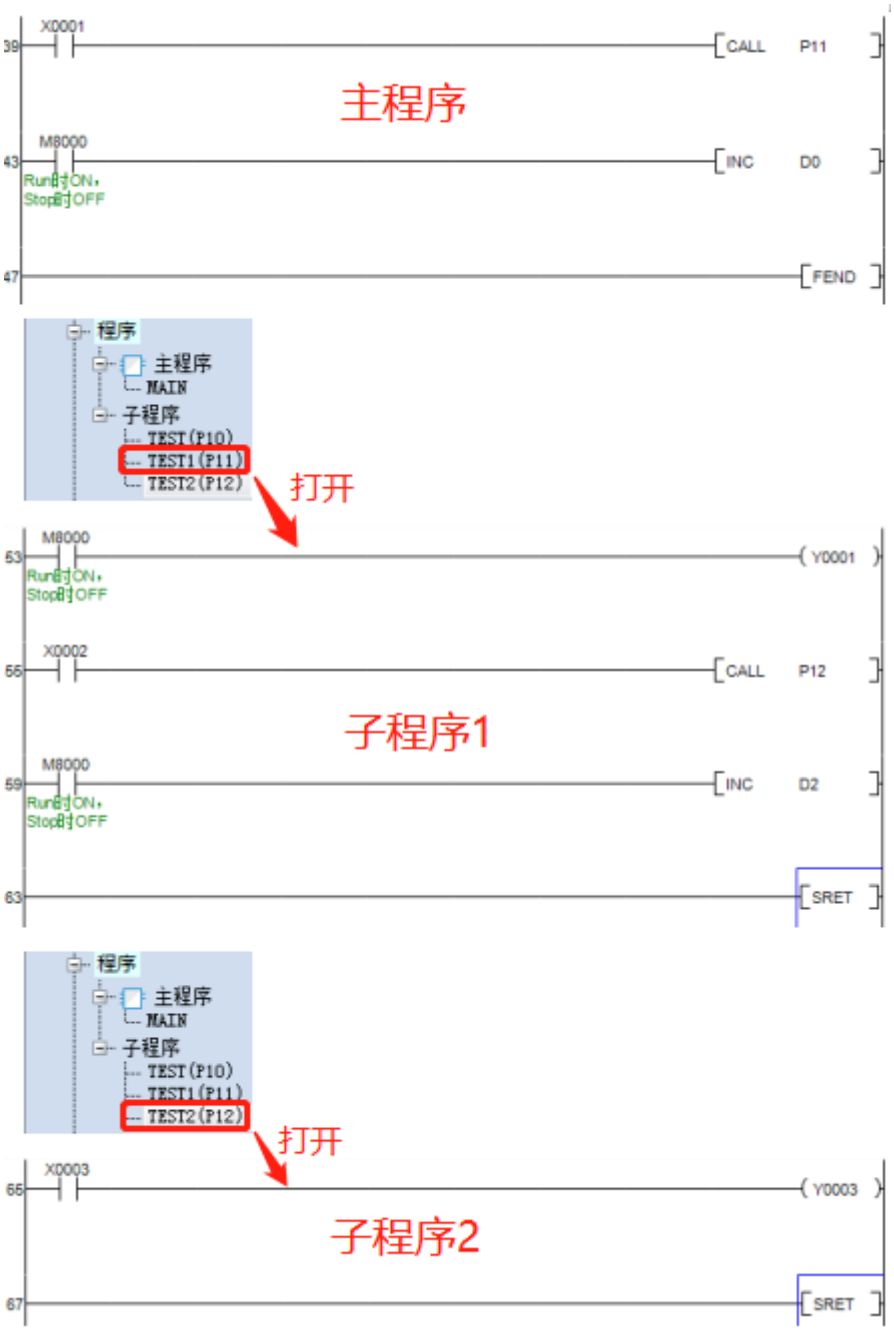


（2）CALL 多层嵌套的使用

主程序：X1=OFF→ON 后仅向标记 P11 跳转一次。

子程序 1:执行 SRET 指令后返回到主程序。执行子程序 1 时，若 X2 为 ON，则向标记 P12 的步跳转。

子程序 2:执行 P12 的子程序后，使用 SRET 指令返回到 P11 的子程序。



4、注意事项

（1）在主程序的最后要用 FEND 指令编程。

（2）CALL 嵌套最多允许 6 层。

3.2.3 SRET 指令(子程序返回)

主要用于 CALL 指令调用后返回，用法参照 CALL 指令中例程。

3.2.4 IRET、EI、DI 指令(中断程序用)

用法说明参照 5 中断程序

3.2.5 FEND 指令(主程序结束)

FEND 指令是主程序结束命令，功能同 END 指令，只是作为主程序专用的结束指令。

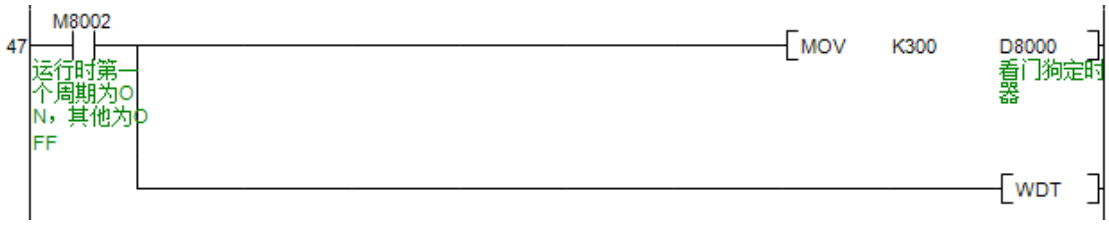
3.2.6 WDT 指令(看门狗)

1、指令格式

2、功能和动作说明

可编程控制器的运算周期(0～END 或 FEND 指令的执行时间)如要超出 200ms（默认值，可修改）时，可编程控制器会出现看门狗定时器错误(检测出运算异常)，然后 ERROR(ERR) LED 灯亮后停止。类似这样的运算周期较长的情况，在程序中间插入 WDT 指令，可以避免出现这样的错误。

3、程序举例



设置看门狗定时器时间 300ms；

看门狗定时器刷新：没有编写 WDT 指令的时候，END 处理时 D8000 的值变为有效。

4、注意事项

避免看门狗定时器超时的方法：

（1）控制梯形图运行周期。

（2）梯形图中 FOR、NEXT 这类流程指令使用次数不要太多。

（3）当梯形图中有运行起来比较耗时的程序时，提前刷新看门狗定时器或者更改看门狗定时器的时间。

3.2.7 FOR、NEXT 指令(循环)

1、指令格式

16bit 3 步		32bit		指令格式
FOR	\	\	\	FOR S

操作数的数据类型如下表

操作数	内容	类型
S	FOR～NEXT 指令之间的重复次数（1~32767，-32768~0 时，作为 1 处理）	BIN16 位

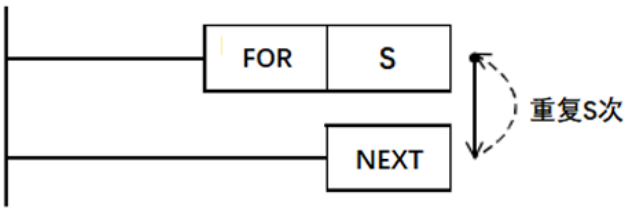
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	

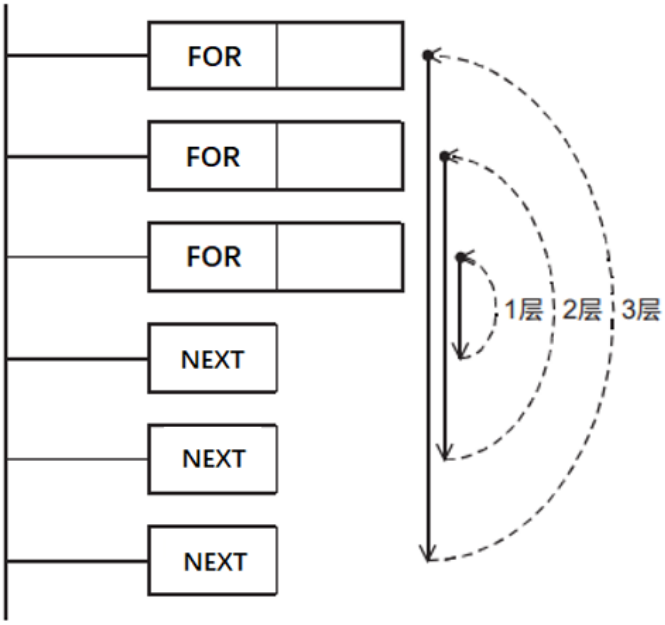
2、功能和动作说明

从 FOR 指令开始到 NEXT 指令之间的程序按指定次数重复运行。

（1）单层 FOR-NEXT



（2）多层 FOR-NEXT 嵌套



3、程序举例

参考上面的功能动作说明。

4、注意事项

（1）FOR-NEXT 嵌套使用最多允许 6 层。

（2）使用多层嵌套时注意看门狗超时时间，可能会引发超时，必要时要刷新看梦时间。

（3）FOR-NEXT 对应个数需匹配。

3.3 传送比较指令

传送比较指令是使用应用指令时最为重要的数据传送和比较等基本的数据操作指令。

3.3.1 CMP 指令(比较)

比较 2 个值，将其结果(大、一致、小)输出到位软元件中(3 点)

1、指令格式

16bit 7 步		32bit 13 步		指令格式
CMP	CMPP	DCMP	DCMPP	CMP S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	成为比较值的数据或软元件编号	BIN16/32 位
S2	成为比较源的数据或软元件编号	BIN16/32 位
D	输出比较结果的起始位软元件编号	位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
D		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
D														

2、功能和动作说明

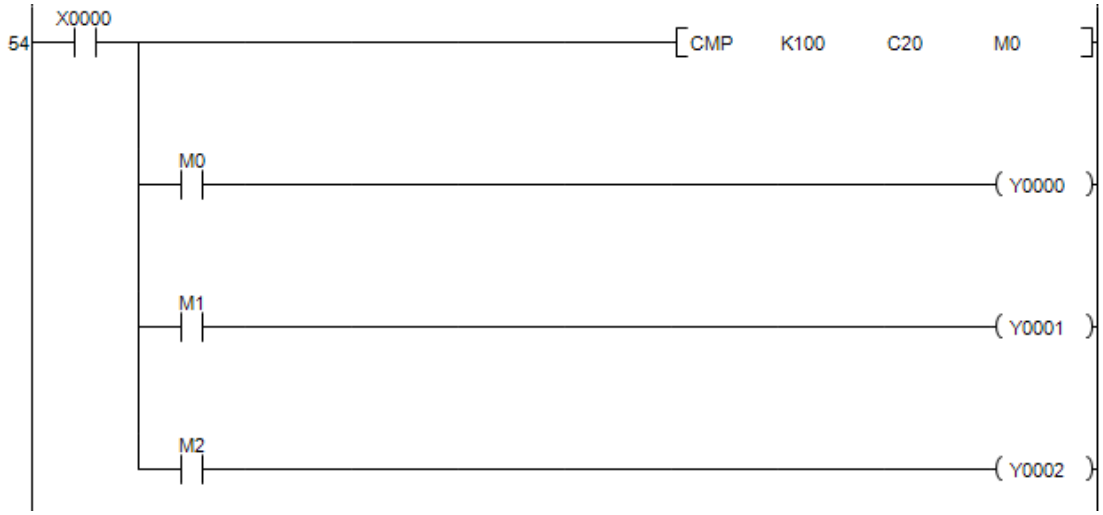
（1）16 位指令

对比较值 S1 和比较源 S2 的内容进行比较，根据其结果(小、一致、大)，使 D、D+1、D+2 中的一个为 ON。

（2）32 位指令

对比较值（S1+1，S1）和比较源（S2+1，S2)的内容进行比较，根据其结果(小、一致、大)，使 D、D+1、D+2 中的一个为 ON。

3、程序举例



C20 当前值 < 100 时，M0 置 ON，Y0 点亮；

C20 当前值 = 100 时，M1 置 ON，Y1 点亮；

C20 当前值 > 100 时，M2 置 ON，Y2 点亮。

4、注意事项

（1）D 中指定的软元件个数是连续的三个，注意不要与其他正在使用的软元件重复。

（2）即使是指令输入为 OFF,CMP 指令不执行时,D～D+2 也会保持当指令输入从 ON 变为 OFF 之前的状态。

3.3.2 ZCP 指令(区间比较)

针对 2 个值(区间)，将与比较源的值比较得出的结果(小于、等于（区域内）、大于)输出到位软元件(3 点)中。

1、指令格式

16bit 9 步		32bit 17 步		指令格式
ZCP	ZCPP	DZCP	DZCPP	ZCP S1 S2 S D

操作数的数据类型如下表

操作数	内容	类型
S1	下侧的比较值的数据或软元件编号	BIN16/32 位
S2	上侧的比较值的数据或软元件编号	BIN16/32 位

S	成为比较源的数据或软元件编号	BIN16/32 位
D	输出比较结果的起始位软元件编号	位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
S											●	●		
D		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D														

2、功能和动作说明

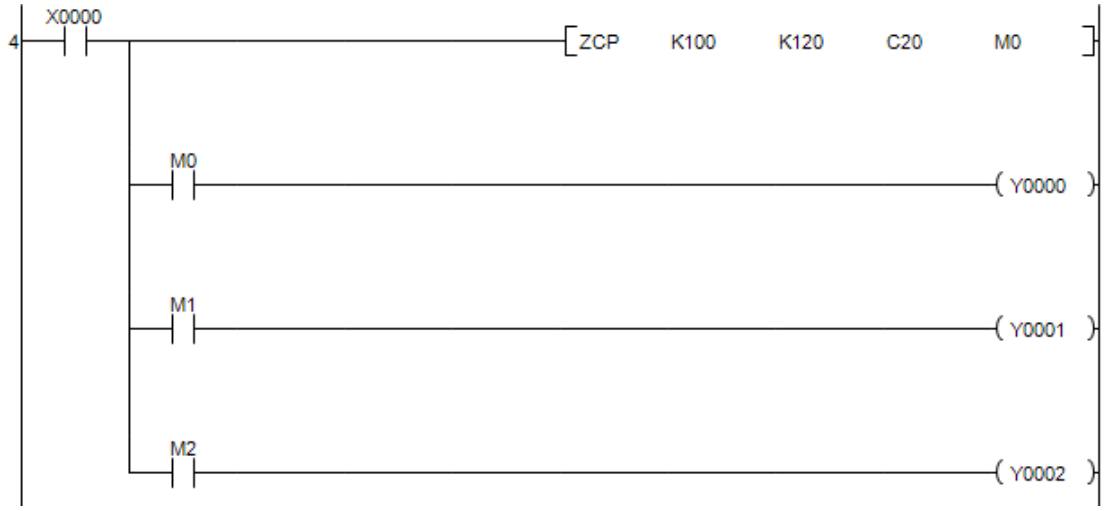
（1）16 位指令

将比较源 S 的内容与下比较值 S1 和上比较值 S2 进行比较，根据其结果(小、区域内、大)，使 D、D+1、D+2 其中一个为 ON。

（2）32 位指令

将比较源（S+1，S）的内容与下比较值（S1+1，S1）和上比较值（S2+1，S2）进行比较，根据其结果(小、区域内、大)，使 D、D+1、D+2 其中一个为 ON。

3、程序举例



C20 当前值 < 100 时，M0 置 ON，Y0 点亮；

100 ≤ C20 当前 ≤ 120 时，M1 置 ON，Y1 点亮；

C20 当前值 > 120 时，M2 置 ON，Y2 点亮。

4、注意事项

（1）D 中指定的软元件个数是连续的三个，注意不要与其他正在使用的软元件重复。

（2）即使是指令输入为 OFF,CMP 指令不执行时，D～D+2 也会保持当指令输入从 ON 变为 OFF 之前的状态。

3.3.3 MOV 指令(传送)

将软元件的内容传送(复制)到其他的软元件中的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
MOV	MOVP	DMOV	DMOVP	MOV S D

操作数的数据类型如下表

操作数	内容	类型
S	传送源的数据，或是保存数据的软元件编号	BIN16/32 位
D	传送目标的软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

（1）16 位指令

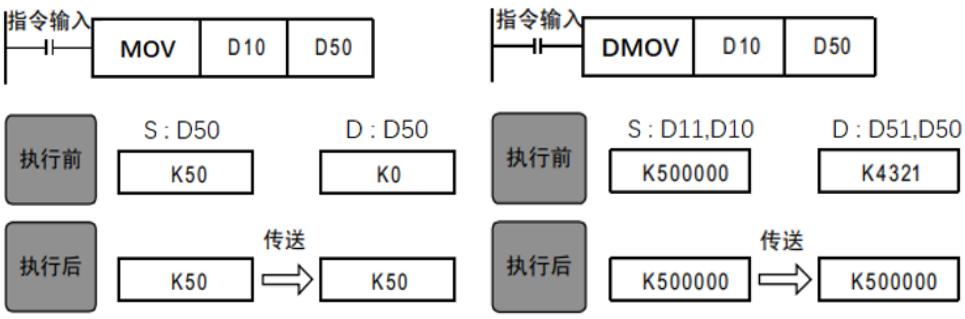
将传送源 S 的内容 1 点传送给传送目标 D，指令输入为 OFF 时，传送目标 D 不变化。传送源 S 中指定了常数(K)时，会自动执行 BIN 转换。

（2）32 位指令

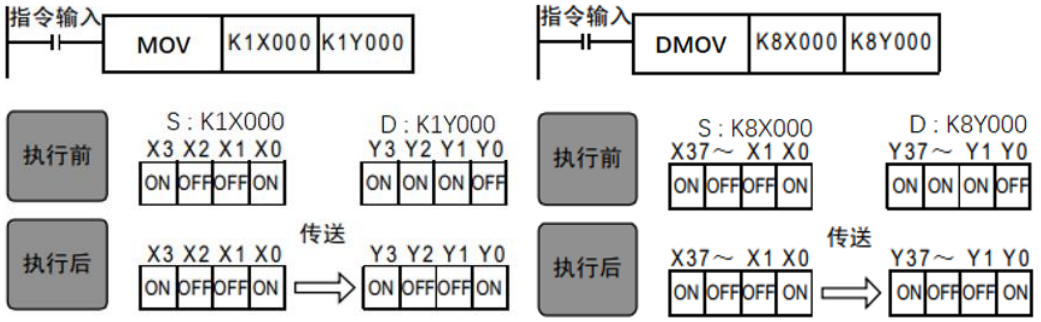
将传送源(S+1,S)的内容 1 点传送给传送目标(D+1,D)，指令输入为 OFF 时，传送目标(D+1,D)不变化。传送源(S+1,S)中指定了常数(K)时，会自动执行 BIN 转换。

3、程序举例

（1）传送字软元件



（2）传送组合位元件



4、注意事项

如果传送的是组合位元件，16 位指令最多传送 16 个（4 的倍数）位软元件，32 位指令最多传送 32 个（4 的倍数）位软元件。

3.3.4 SMOV 指令(位移动)

以位数为单位(4 位)进行数据分配合成的指令。

1、指令格式

16bit 11 步		32bit		指令格式
SMOV	SMOVP	\	\	SMOV S m1 m2 D n

操作数的数据类型如下表

操作数	内容	类型
S	保存有要进行位移动的数据软元件的编号	BIN16 位
m1	要移动的起始位的位置	BIN16 位
m2	要移动的位的个数	BIN16 位
D	保存已经进行位移动的数据的软元件编号	BIN16 位
n	移动目标的起始位的位置	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														

m1											●	●		
m2											●	●		
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
m1														
m2														
D		●	●	●	●	●	●	●	●	●	●	●	●	
n														

2、功能和动作说明

传送源 S 和传送目标 D 的内容转换(0000～9999)成 4 位数的 BCD，m1 位数起的低 m2 位数部分被传送(合成)到传送目标 D 的 n 位数起始处，然后转换成 BIN，保存在传送目标 D 中。

3、程序举例

4、注意事项

（1）原数据取值范围是 0~9999。

（2）m1，m2，n 的范围是 1~4。

（3）将 M8168 置 ON 后，执行 SMOV 指令时，则不能进行 BIN→BCD 转换。位移动以 4 位为单位执行。

3.3.5 CML 指令(反转传送)

以位为单位反转数据后进行传送(复制)的指令

1、指令格式

16bit 5 步		32bit 9 步		指令格式
CML	CMLP	DCML	DCMLP	CML S D

操作数的数据类型如下表

操作数	内容	类型
S	要执行反转的数据，或是保存数据的字软元件编号	BIN16/32 位
D	保存要执行反转后的数据的目标字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

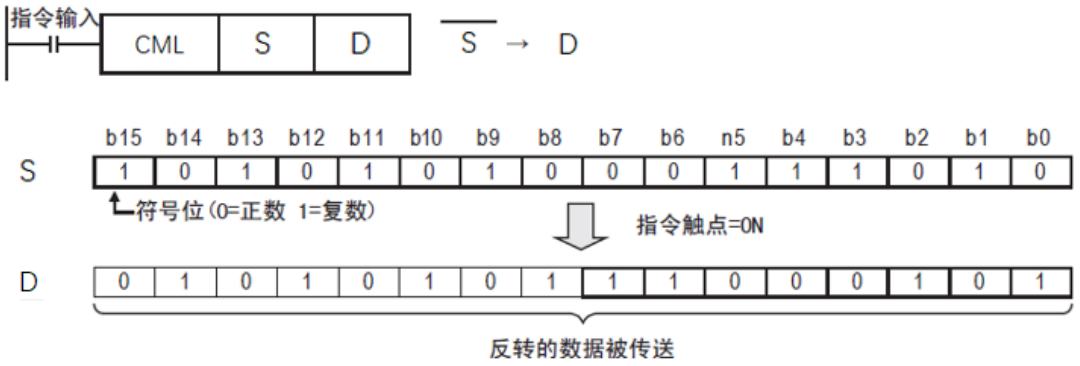
（1）16 位指令

将 S 中指定的软元件的各位反转(0→1,1→0)后，传送至 D。

（2）32 位指令

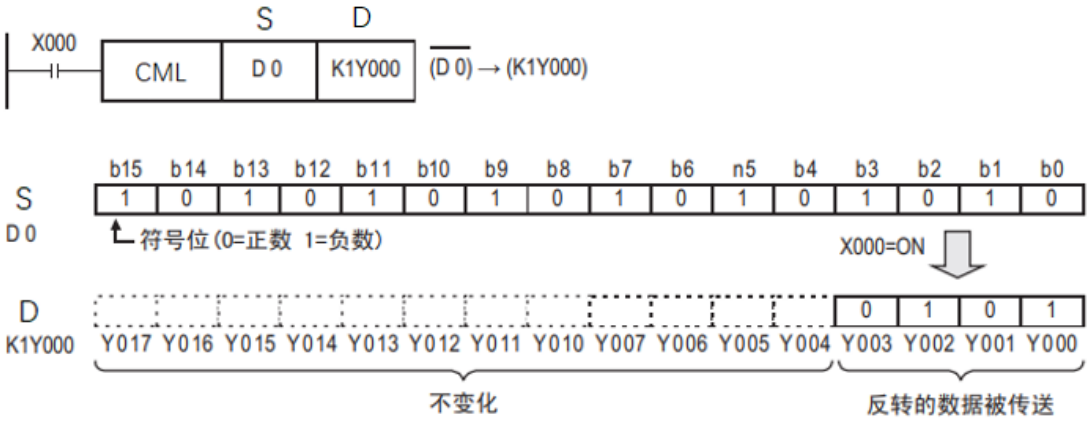
将（S+1，S）中指定的软元件的各位反转(0→1,1→0)后，传送至（D+1，D）。

3、程序举例



4、注意事项

指定的是位组合元件时，根据实际的位数来反转传送，但是这个位数不能超过指令位数且为 4 的倍数，下面的例子是指定位数为 4 的情况。



3.3.6 BMOV 指令(成批传送)

对指定点数的多个数据进行成批传送(复制)。

1、指令格式

16bit 7 步		32bit		指令格式
BMOV	BMOVP	\	\	BMOV S D n

操作数的数据类型如下表

操作数	内容	类型
S	传送源的数据，或是保存数据的软元件编号	BIN16

D	传送目标的软元件编号	BIN16
n	传送点数(包括文件寄存器)[n≦512]	BIN16

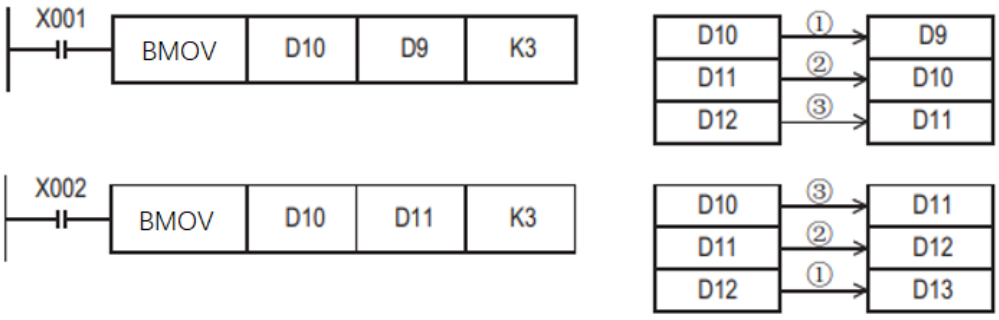
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	
n								●	●	●				

2、功能和动作说明

将 S 开始的 n 点的数据成批传送到 D 开始的 n 点中。超出软元件编号范围时，在可能的范围内传送。

3、程序举例

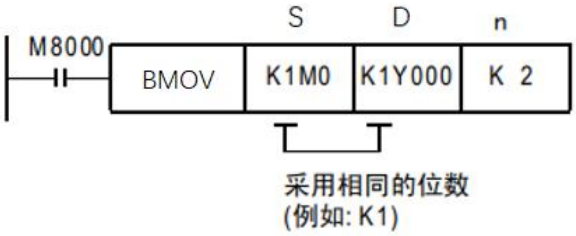


4、注意事项

（1）无论传送源的数据有无传送，为了防止数据源没有传送就被改写，采用编号重叠的方法，按上面的①～③的顺序自动传送。

（2）双向传送功能，当 M8024 标志位置 ON 时，传送方向变为 D->S。

（3）带有位数指定的位软元件的情况下，S 和 D 要采用相同的位数



3.3.7 FMOV 指令(多点传送)

将同一数据传送到指定点数的软元件中的进行多点传送指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
FMOV	FMOVP	DFMOV	DFMOVP	FMOV S D n

操作数的数据类型如下表

操作数	内容	类型
S	传送源的数据，或是保存数据的软元件编号	BIN16/32 位
D	传送目标的起始字软元件编号(传送源的同一数据被成批传送)	BIN16/32 位
n	传送点数[K1≦n≦K512，H1≦n≦H1FF]	BIN16

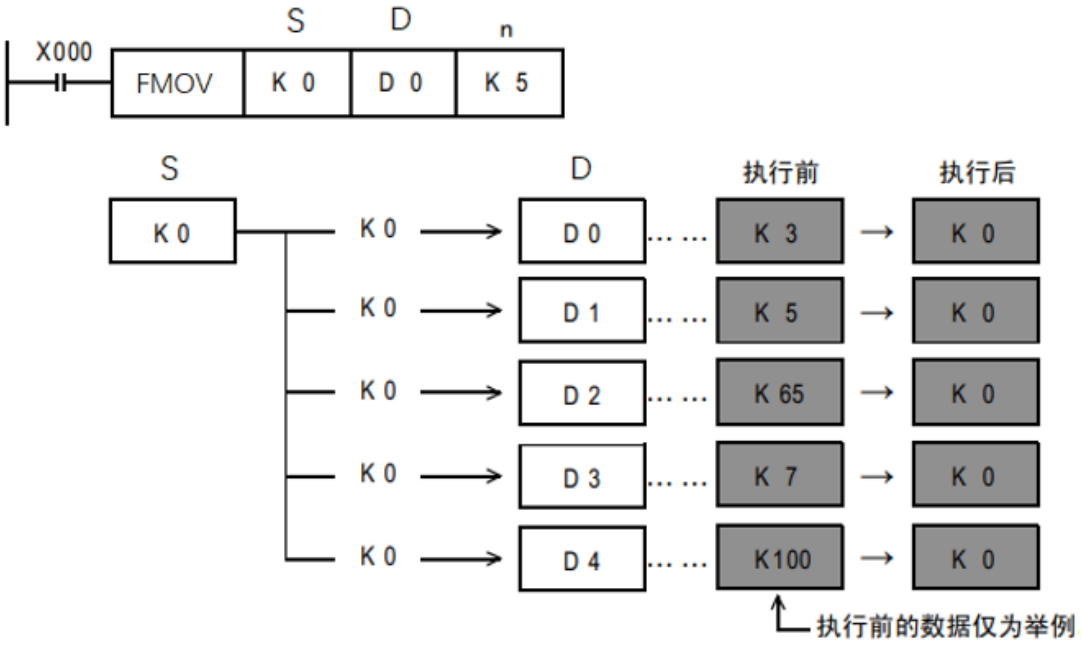
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●			●	
n														

2、功能和动作说明

将 S 的内容传送到以 D 起始的 n 点的软元件中，n 点的软元件内容都相同，以 n 指定的个数超出了软元件编号范围时，在可能的范围内传送。

3、程序举例



4、注意事项

n 指定的个数超出了软元件编号范围时，在可能的范围内传送。

3.3.8 XCH 指令(交换)

在 2 个软元件之间进行数据交换

1、指令格式

16bit 5 步		32bit 9 步		指令格式
XCH	XCHP	DXCH	DXCHP	XCH D1 D2

操作数的数据类型如下表

操作数	内容	类型
D1	保存交换数据的软元件编号	BIN16/32 位
D2	保存交换数据的软元件编号	BIN16/32 位

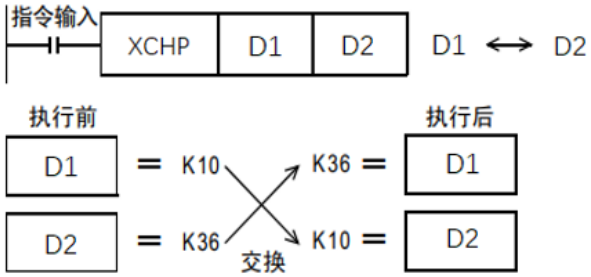
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D1														
D2														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D1		●	●	●	●	●	●	●	●	●	●	●	●	
D2		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

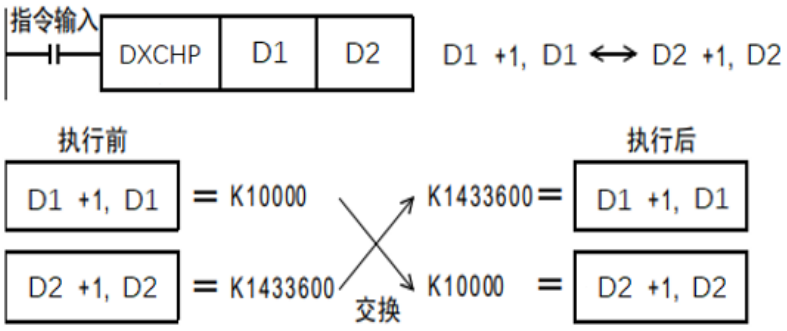
（1）16 位指令

D1 和 D2 相互之间进行数据交换。

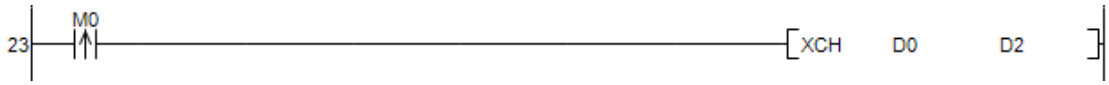


（2）32 位指令

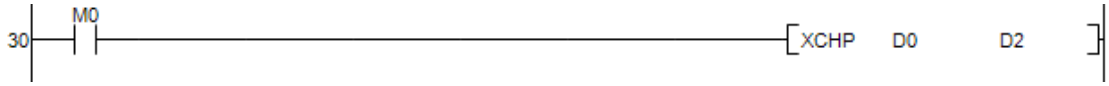
(D1+1，D1)和(D2+1,D2)相互之间进行数据交换。



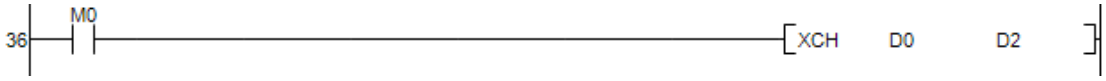
3、程序举例



或



当 M0 接通时，D0 与 D2 的值进行交换（执行一次交换）

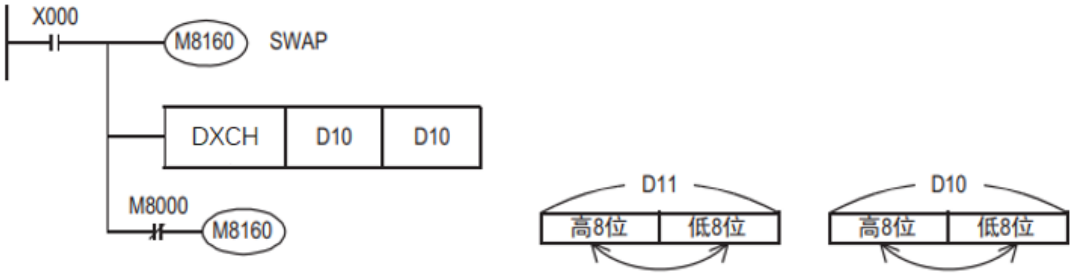


当 M0 接通时，D0 与 D2 每个扫描周期都会交换一次，使用时需注意。

4、注意事项

（1）在 M8160 为 ON 的状态下执行指令时，交换字软元件的高 8 位(字节)和低 8 位(字节)。该动作和 SWAP 指令为相同的动作。

（2）32 位运算时，交换各个字软元件的高 8 位(字节)和低 8 位(字节)。



（3）M8160 为 ON 时，D1 和 D2 的软元件编号需要保持一致。

3.3.9 BCD 指令(BCD 转换)

将 BIN(2 进制数)转换成 BCD(10 进制数)后传送的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
BCD	BCDP	DBCD	DBCDP	BCD S D

操作数的数据类型如下表

操作数	内容	类型
S	保存转换源(2 进制数)数据的字软元件编号	BIN16/32 位
D	转换目标(10 进制数)的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D1														
D2														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D1	●	●	●	●	●	●	●	●	●	●	●	●	●	
D2		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

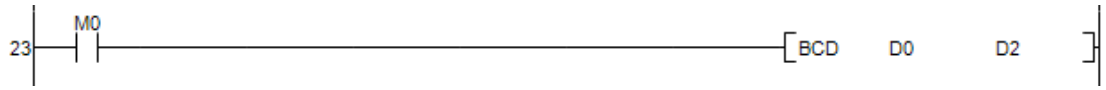
（1）16 位指令

将 S 的 BIN(2 进制数)数据转换成 BCD(10 进制数)数据后传送到 D 中。S 的数据可以转换成 K0～K9999 的 BCD(10 进制数)。

（2）32 位指令

将（S+1，S）的 BIN(2 进制数)数据转换成 BCD(10 进制数)数据后传送到（D+1，D）中。（S+1，S）的数据，可以转换成 K0～K99999999 的 BCD(10 进制数)。

3、程序举例



当 M0 接通时，D0 中的 BIN(2 进制数)数据转换成 BCD(10 进制数)数据后传送到 D2 中。

4、注意事项

注意 16 位和 32 位指令使用时，源操作数的取值范围。16 位是 0~9999,32 位是 0~99999999。

3.3.10 BIN 指令(BIN 转换)

将 10 进制数(BCD)转换成 2 进制数(BIN)的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
BIN	BINP	DBIN	DBINP	BIN S D

操作数的数据类型如下表

操作数	内容	类型
S	保存转换源(10 进制数)数据的字软元件编号	BIN16/32 位
D	转换目标(2 进制数)的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

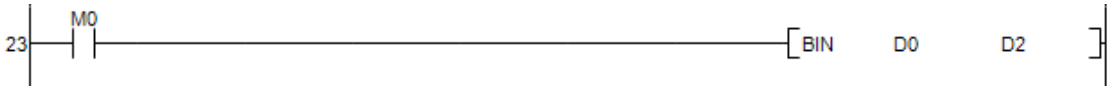
（1）16 位指令

将 S 的 BCD(10 进制数)数据转换成 BIN(2 进制数)数据后传送到 D 中。S 的数据可以在 0～9999(BCD)的范围内转换。

（2）32 位指令

将(S+1，S)的 BCD(10 进制数)数据转换成 BIN(2 进制数)数据后传送到(D+1，D)中。(S+1，S)的数据，可以在 0～99999999(BCD)的范围内转换。

3、程序举例



当 M0 接通时，D0 中的 BCD(10 进制数)数据转换成 BIN(2 进制数)数据后传送到 D2 中。

4、注意事项

注意源操作数的取值范围。

3.4 四则逻辑运算指令

3.4.1 ADD 指令(BIN 加法运算)

2 个值进行加法运算(A+B=C)后得出结果的指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
ADD	ADDP	DADD	DADDP	ADD S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	加法运算的数据，或是保存数据的字软件编号	BIN16/32 位
S2	加法运算的数据，或是保存数据的字软件编号	BIN16/32 位
D	保存加法运算结果的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

（1）16 位指令

将 S1 和 S2 的内容进行二进制加法运算后传送到 D 中。

（2）32 位指令

将（S1+1，S1）和（S2+1，S2）的内容进行二进制加法运算后传送到（D+1，D）中。

2、程序举例

当 M0 接通时，将 D0 和 D2 的内容进行二进制加法运算后传送到 D4 中。

4、注意事项

标志位的动作及数值的正负的关系：

软元件	名称	内容
M8020	零位	ON：运算结果为 0 时 OFF：运算结果为 0 以外时
M8021	借位	ON：运算结果小于-32,768(16 位运算)或是-2,147,483,648(32 位运算)时，借位标志位动作。 OFF：运算结果不小于-32,768(16 位运算)或是-2,147,483,648(32 位运算)时
M8022	进位	ON：运算结果大于 32,767(16 位运算) 或者 2,147,483,647(32 位运算) 时，进位标志位动作。 OFF：运算结果不大于 32,767(16 位运算)或者 2,147,483,647(32 位运算)时

3.4.2 SUB 指令(BIN 减法运算)

2 个值进行减法运算(A－B=C)后得出结果的指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
SUB	SUBP	DSUB	DSUBP	SUB S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	减法运算的数据，或是保存数据的字软件编号	BIN16/32 位
S2	减法运算的数据，或是保存数据的字软件编号	BIN16/32 位
D	保存减法运算结果的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

（1）16 位指令

将 S1 和 S2 的内容进行二进制减法运算后传送到 D 中。

（2）32 位指令

将（S1+1，S1）和（S2+1，S2）的内容进行二进制减法运算后传送到（D+1，D）中。

3、程序举例

当 M1 接通时，将 D0 和 D2 的内容进行二进制减法运算后传送到 D4 中。

4、注意事项

标志位的动作及数值的正负的关系：

软元件	名称	内容
M8020	零位	ON：运算结果为 0 时 OFF：运算结果为 0 以外时
M8021	借位	ON：运算结果小于-32,768(16 位运算)或是-2,147,483,648(32 位运算)时，借位标志位动作。 OFF：运算结果不小于-32,768(16 位运算)或是-2,147,483,648(32 位运算)时
M8022	进位	ON：运算结果大于 32,767(16 位运算) 或者 2,147,483,647(32 位运算) 时，进位标志位动作。 OFF：运算结果不大于 32,767(16 位运算)或者 2,147,483,647(32 位运算)时

3.4.3 MUL 指令(BIN 乘法运算)

2 个值进行乘法运算(A×B=C)后得出结果的指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
MUL	MULP	DMUL	DMULP	MUL S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	乘法运算的数据，或是保存数据的字软件编号	BIN16/32 位
S2	乘法运算的数据，或是保存数据的字软件编号	BIN16/32 位
D	保存乘法运算结果的字软元件编号	BIN32/64 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	

D		●	●	●	●	●	●	●	●	●	●	●	●	

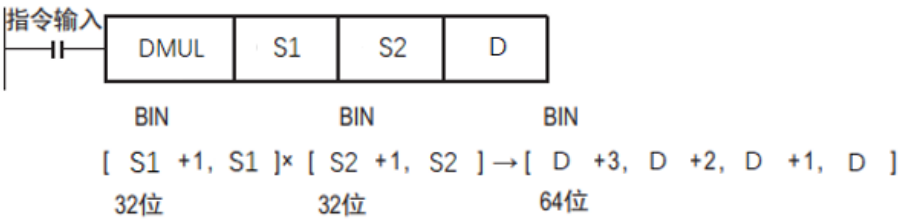
2、功能和动作说明

（1）16 位指令

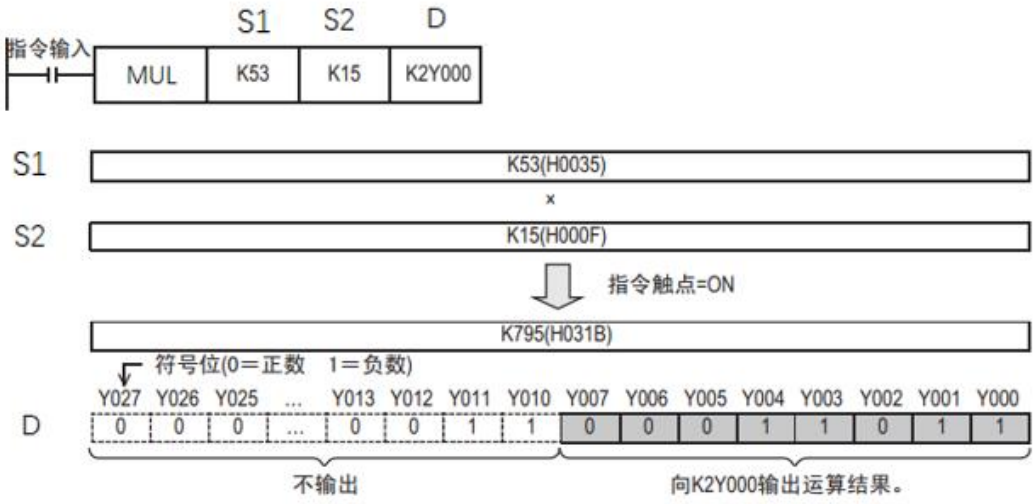
将 S1 和 S2 的内容进行二进制乘法运算后传送到(D+1，D)的 32 位(双字)中。

（2）32 位指令

将(S1+1,S1)和(S2+1,S2)的内容进行二进制乘法运算后传送到(D+3,D+2,D+1,D)的 64 位(四个连续的字软元件)中。



3、程序举例



（D+1，D）指定位数(K1～8)时，例如，指定 K2 时，只能得到乘积(32 位)中的低 8 位。

4、注意事项

（1）使用组合位时，得到的运算结果为位数 x4。

（2）当运算结果为 0 时，M8304 置 ON，其他结果为 OFF。

（3）进行 32 位指令运算时，目的操作数不能为 Z 寄存器。

（4）32 位指令计算出来得到的 64 位数据上位机并不能直接监控，只能通过多个寄存器自己组合出结果，要想更直观的显示，建议使用浮点数运算。

3.4.4 DIV 指令(BIN 除法运算)

2 个值进行加法运算(A/B=C)后得出结果的指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
DIV	DIVP	DDIV	DDIVP	DIV S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	除法运算的数据，或是保存数据的字软件编号	BIN16/32 位
S2	除法运算的数据，或是保存数据的字软件编号	BIN16/32 位
D	保存除法运算结果的字软元件编号	BIN32/64 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

（1）16 位指令

将 S1 和 S2 的内容进行二进制除法运算后，商传送到 D 中，余数传送到 D+1 中。

（2）32 位指令

将（S1+1，S1）和(S2+1,S2)的内容进行二进制除法运算后，商传送到(D+1,D)中，余数传送到(D+3,D+2)中。

3、程序举例

当 M1 接通时，将 D0 和 D2 的内容进行二进制除法运算后，商传送到 D4 中，余数传送到 D5 中。

4、注意事项

（1）有关运算结果：商和余数的最高位显示位正(0)、负(1)的符号。当被除数或除数其一为负时，商为负。当被除数为负时，余数为负。

（2）通过指定位数来指定位软元件时，不能得出余数。使用 32 位运算(DDIV、DDIVP)时，不能指定使用 Z 寄存器。

（3）除数 S2 为 0 时，发生运算错误，运算结果超出指令位数对应的值时，也会报错，且进位标志置 ON。

软元件	名称	内容
M8304	零位	ON：运算结果为 0 时 OFF：运算结果为 0 以外时
M8306	进位	ON ：运算结果超过 32,767(16 位运算)或者 2,147,483,647(32 位运算)时，进位标志位动作。 OFF ：运算结果为 32,767(16 位运算)或者 2,147,483,647(32 位运算)以下时

3.4.5 INC 指令(BIN 加 1)

指定的软元件数据中加“1”(+1 加法)的指令。

1、指令格式

16bit 3 步		32bit 5 步		指令格式
INC	INCP	DINC	DINCP	INC D

操作数的数据类型如下表

操作数	内容	类型
D	保存被加一数据的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

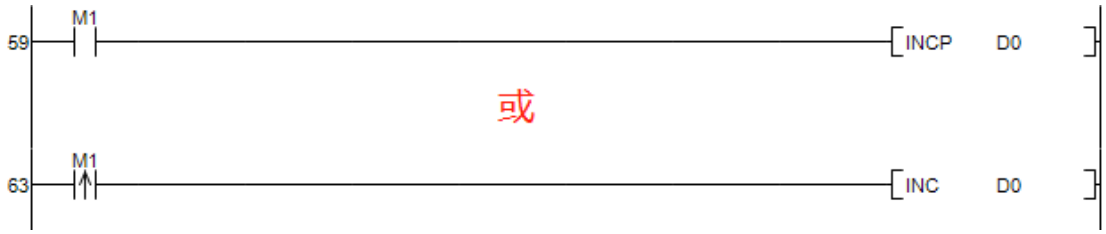
（1）16 位指令

将 D 的内容加一后传送到 D 中。

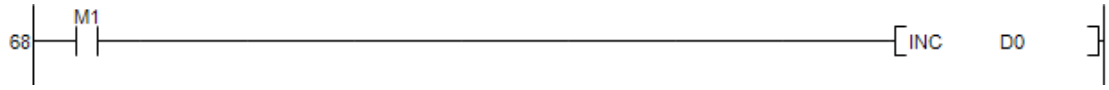
（2）32 位指令

将（D+1，D）的内容加一后传送到（D+1，D）中。

3、程序举例



当 M1 接通时，D0 自加 1(执行一次，下一次 M1 OFF-ON 时再执行一次)。



当 M1 接通时，D0 自加 1(每个扫描周期执行一次，即每个扫描周期自加 1 一次)。

4、注意事项

（1）16 位运算：+32,767 上加 1 后，变为-32,768，但是标志位(零、借位、进位)不动作。

（2）32 位运算：+2,147,483,647 上加 1 后，变为-2,147,483,648，但是标志位(零、借位、进位)不动作。

3.4.6 DEC 指令(BIN 减 1)

指定的软元件数据中减“1”(-1 加法)的指令。

1、指令格式

16bit 3 步		32bit 5 步		指令格式
DEC	DECP	DDEC	DDECP	DEC D

操作数的数据类型如下表

操作数	内容	类型
D	保存被减一数据的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件	常数	实数	字符串

	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

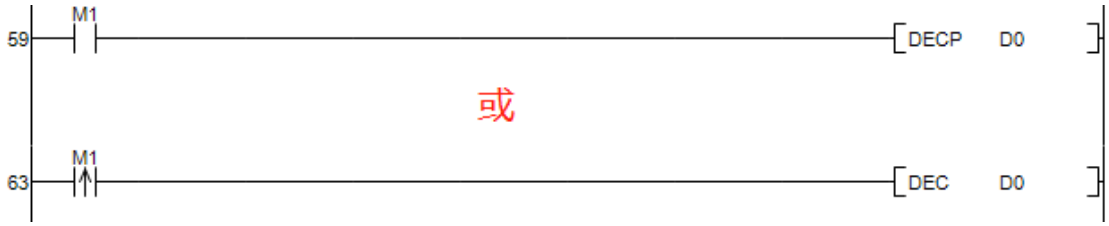
（1）16 位指令

D 的内容减一运算后，传送到 D 中。

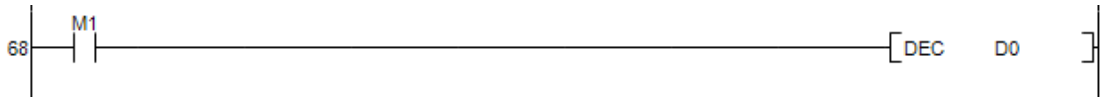
（2）32 位指令

（D+1，D）的内容减一运算后，传送到（D+1，D）中。

3、程序举例



当 M1 接通时，D0 自减 1(执行一次，下一次 M1 OFF-ON 时再执行一次)。



当 M1 接通时，D0 自减 1(每个扫描周期执行一次，即每个扫描周期自减 1 一次)。

4、注意事项

（1）16 位运算：-32,768 上减一后，变为+32,767，但是标志位(零、借位、进位)不动作。

（2）32 位运算：-2,147,483,648 上减一后，变为+2,147,483,647，但是标志位(零、借位、进位)不动作。

3.4.7 WAND 指令(逻辑与)

2 个数值进行逻辑与运算的(AND)指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
WAND	WANDP	DAND	DANDP	WAND S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	逻辑与数据或保存数据的字软元件编号	BIN16/32 位
S2	逻辑与数据或保存数据的字软元件编号	BIN16/32 位
D	保存逻辑与结果的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

（1）16 位指令

S1 和 S2 的内容以各位为单位进行逻辑与(AND)运算后，传送到 D 中。

（2）32 位指令

（S1+1，S1）和（S2+1，S2）的内容以各位为单位进行逻辑与(AND)运算后，传送到（D+1，D）中。

3、程序举例

D0	1124	0 0 1 0 0 1 1 0 0 0 1 0 0 0 0 0
D1	2222	0 1 1 1 0 1 0 1 0 0 0 1 0 0 0 0
D2	36	0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0

当 M2 接通时，D0 和 D1 的内容以各位为单位进行逻辑与(AND)运算后，传送到 D2 中。

4、注意事项

逻辑与运算以位为单位，如下表中(1∧1=1　0∧1=0　1∧0=0　0∧0=0)所示变化。

	S1	S2	D
位单位的逻辑运算	0	0	0
	1	0	0
	0	1	0
	1	1	1

3.4.8 WOR 指令(逻辑或)

2 个数值进行逻辑或运算的(OR)指令。

1、指令格式

16bit　7 步		32bit　13 步		指令格式
WOR	WORP	DOR	DORP	WOR S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	逻辑或数据或保存数据的字软元件编号	BIN16/32 位
S2	逻辑或数据或保存数据的字软元件编号	BIN16/32 位
D	保存逻辑或结果的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

（1）16 位指令

S1 和 S2 的内容以各位为单位进行逻辑或(OR)运算后，传送到 D 中。

（2）32 位指令

（S1+1，S1）和（S2+1，S2）的内容以各位为单位进行逻辑或(OR)运算后，传送到（D+1，D）中。

3、程序举例

参考 2.4.7 WAND 指令例程。

4、注意事项

逻辑或运算以位为单位，如下表中(1∨1=1　0∨1=1　0∨0=0　1∨0=1)所示变化。

	S1	S2	D
位单位的逻辑运算	0	0	0
	1	0	1
	0	1	1
	1	1	1

3.4.9 WXOR 指令(逻辑异或)

2 个数值进行逻辑异或运算的(XOR)指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
WXOR	WXORP	DXOR	DXORP	WXOR S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	逻辑异或数据或保存数据的字软元件编号	BIN16/32 位
S2	逻辑异或数据或保存数据的字软元件编号	BIN16/32 位
D	保存逻辑异或结果的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

（1）16 位指令

S1 和 S2 的内容以各位为单位进行逻辑异或(XOR)运算后，传送到 D 中。

（2）32 位指令

（S1+1，S1）和（S2+1，S2）的内容以各位为单位进行逻辑异或(XOR)运算后，传送到（D+1，D）中。

3、程序举例

参考 2.4.7 WAND 指令例程。

4、注意事项

逻辑异或运算以位为单位，如下表中(1∀1=0　0∀0=0　1∀0=1　0∀1=1)所示变化。

	S1	S2	D
位单位的逻辑运算	0	0	0
	1	0	1
	0	1	1
	1	1	1

3.4.10 NEG 指令(补码)

求出数值的 2 进制补码(各位反转+1 后的值)的指令。使用这个指令后，可以反转数值的符号。

1、指令格式

16bit 3 步		32bit 5 步		指令格式
NEG	NEGP	DNEG	DNEGP	NEG S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
D	保存欲求补码的数据的字软元件编号，以及保存目标软元件编号(运算结果被保存在同一字软元件编号中。)	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

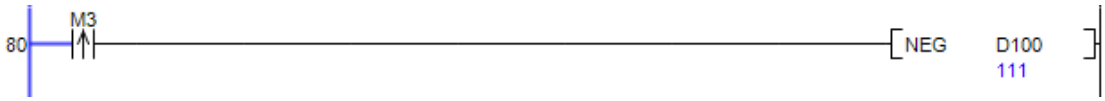
（1）16 位指令

将 D 内容中的各位反转(0→1、1→0)后加 1 的结果保存到原先的软元件中。

（2）32 位指令

将(D+1，D)内容中的各位反转(0→1、1→0)后加一的结果保存到原先的软元件中。

3、程序举例



执行前	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	111
按位反转	0	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	
加1	1	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	-111

低位 → 高位

4、注意事项

使用连续执行型(NEG、DNEG)指令时，每个扫描周期(各运算周期)都执行，请务必注意。

3.5 循环移位指令

3.5.1 ROR 指令(循环右移)

使不包括进位标志在内的指定位数部分的位信息右移、循环的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
ROR	RORP	DROR	DRORP	ROR D n

操作数的数据类型如下表

操作数	内容	类型
D	保存循环右移数据的字软元件编号	BIN16/32 位
n	循环移动的位数[n≦16(16 位指令)，n≦32(32 位指令)]	BIN16/32 位

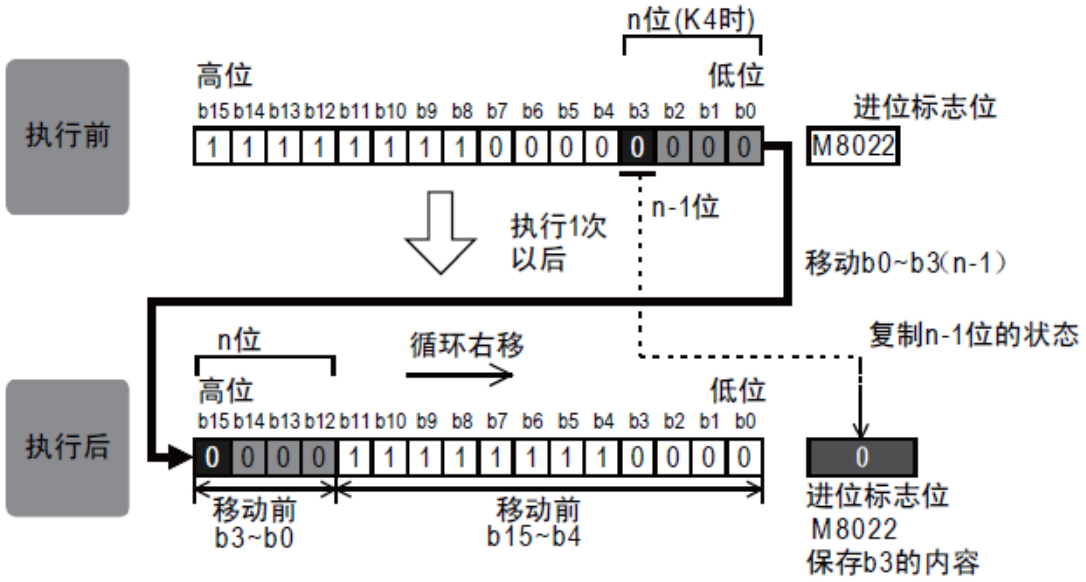
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●	●	●	●	
n							●	●	●	●				

2、功能和动作说明

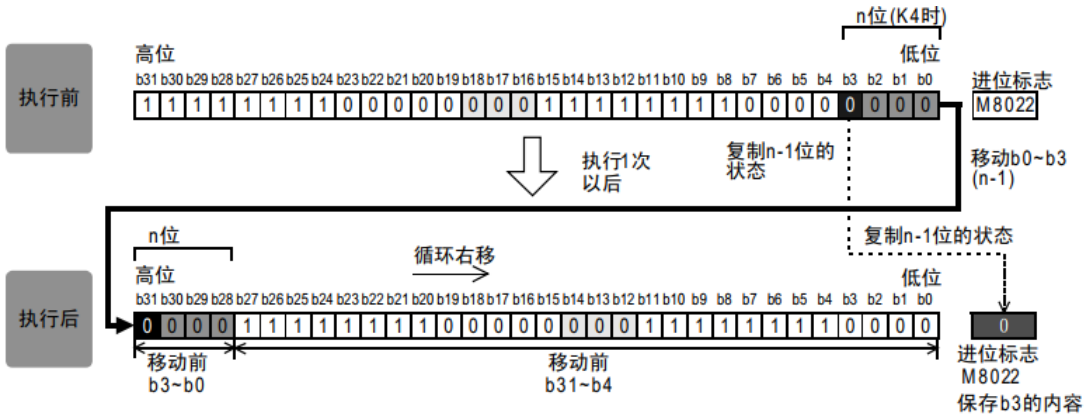
（1）16 位指令

D 的 16 位中的 n 位循环右移。最后的位保存在进位标志位(M8022)中。位数指定软元件的情况下，仅 K4(16 位指令)有效。



（2）32 位指令

将（D+1，D）的 32 位中的 n 位循环右移。最后的位保存在进位标志位(M8022)中。位数指定软元件的情况下，仅 K8(32 位指令)有效。



3、程序举例

参考功能和动作说明。

4、注意事项

（1）相关软元件：M8022，进位标志，最后从最低位移出的位为 1 时置 ON。

（2)连续执行型(ROR、DROR)指令的场合请注意每个扫描周期(运算周期)都会执行循环移位。

（3)在 D 中指定位数指定软元件时，仅 K4(16 位指令)或 K8(32 位指令)有效。(例如 K4Y010、K8M0) 。

3.5.2 ROL 指令(循环左移)

使不包括进位标志位在内的指定位数部分的位信息左移、循环的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
ROL	ROLP	DROL	DROLP	ROL D n

操作数的数据类型如下表

操作数	内容	类型
D	保存循环左移数据的字软元件编号	BIN16/32 位
n	循环移动的位数[n≦16(16 位指令)，n≦32(32 位指令)]	BIN16/32 位

操作数的对象软元件如下表

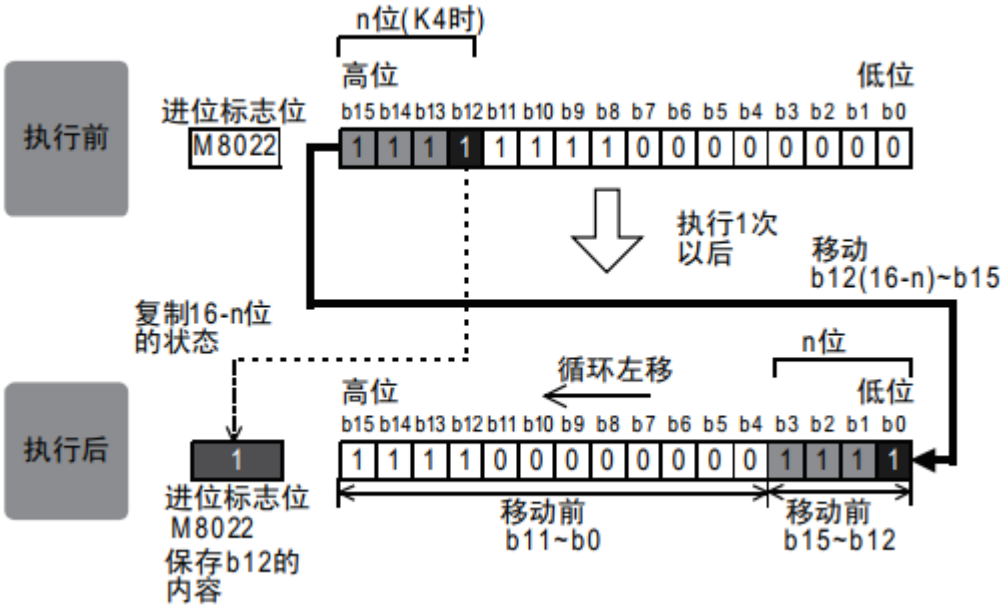
操作数种类	位软元件	常数	实数	字符串

	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●	●	●	●	
n							●	●	●	●				

2、功能和动作说明

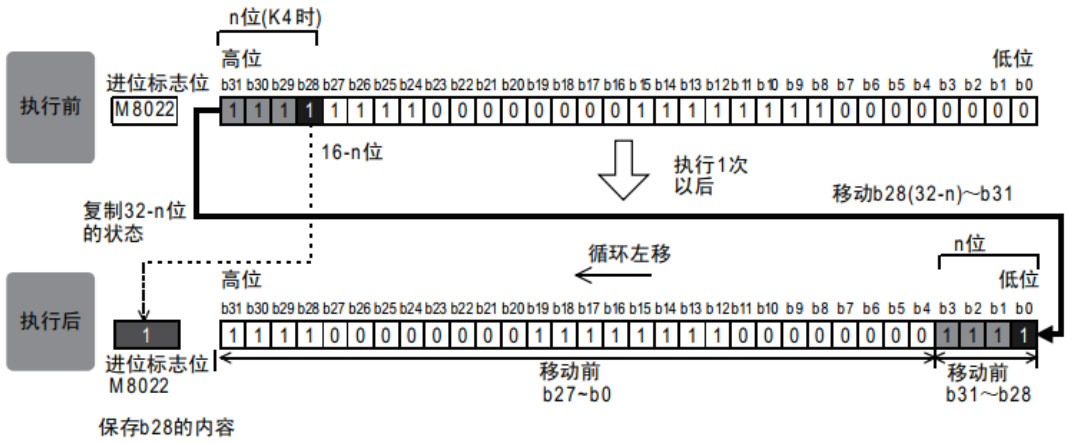
（1）16 位指令

D 的 16 位中的 n 位循环左移。最后的位保存在进位标志位(M8022)中。位数指定软元件的情况下，仅 K4(16 位指令)有效



（2）32 位指令

（D+1，D）的 16 位中的 n 位循环左移。最后的位保存在进位标志位(M8022)中。位数指定软元件的情况下，仅 K4(16 位指令)有效



3、程序举例

参考功能和动作说明。

4、注意事项

（1）相关软元件：M8022，进位标志，最后从最高位移出的位为 1 时置 ON。

（2)连续执行型(ROL、DROL)指令的场合请注意每个扫描周期(运算周期)都会执行循环移位。

（3)在 D 中指定位数指定软元件时，仅 K4(16 位指令)或 K8(32 位指令)有效。(例如 K4Y010、K8M0) 。

3.5.3 RCR 指令(带进位循环右移)

使不包括进位标志位在内的指定位数部分的位信息右移、循环的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
RCR	RCRP	DRCR	DRCRP	RCR D n

操作数的数据类型如下表

操作数	内容	类型
D	保存循环右移数据的字软元件编号	BIN16/32 位
n	循环移动的位数[n≦16(16 位指令)，n≦32(32 位指令)]	BIN16/32 位

操作数的对象软元件如下表

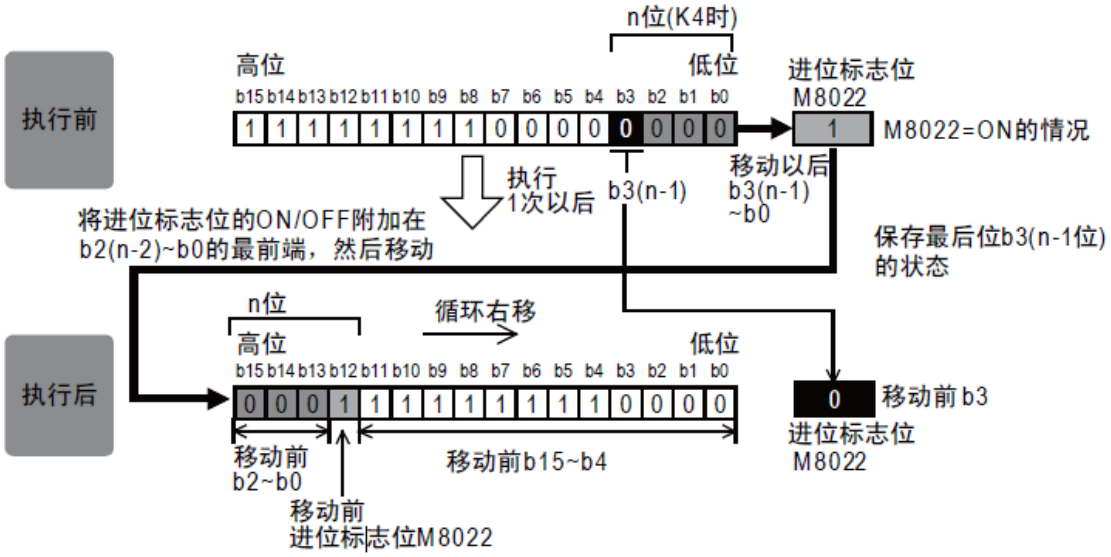
操作数种类	位软元件	常数	实数	字符串

	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●	●	●	●	
n							●	●	●	●				

2、功能和动作说明

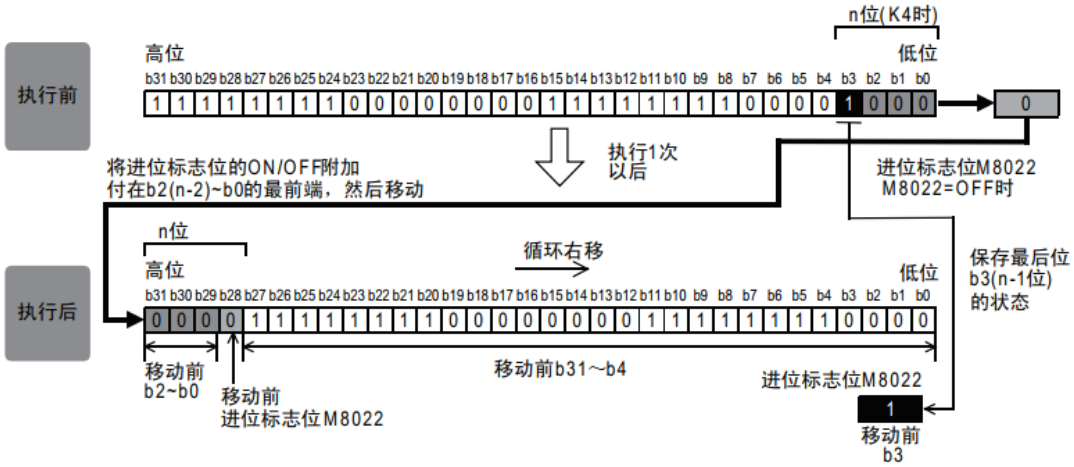
（1）16 位指令

D 的 16 位+1 位(进位标志位 M8022)向右移动 n 位。因为循环回路中有进位标志位，所以如果执行循环移位指令之前 M8022 就先 ON 或 OFF，则会被送入目标操作数中。



（2）32 位指令

（D+1，D）的 32 位+1 位(进位标志位 M8022)向右移动 n 位。



3、程序举例

参考功能和动作说明。

4、注意事项

（1）相关软元件：M8022，进位标志，最后从最低位移出的位为 1 时置 ON。

（2)连续执行型(RCR、DRCR)指令的场合请注意每个扫描周期(运算周期)都会执行循环移位。

（3)在 D 中指定位数指定软元件时，仅 K4(16 位指令)或 K8(32 位指令)有效。(例如 K4Y010、K8M0) 。

3.5.4 RCL 指令(带进位循环左移)

使不包括进位标志位在内的指定位数部分的位信息左移、循环的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
RCL	RCLP	DRCL	DRCLP	RCL D n

操作数的数据类型如下表

操作数	内容	类型
D	保存循环左移数据的字软元件编号	BIN16/32 位
n	循环移动的位数[n≦16(16 位指令)，n≦32(32 位指令)]	BIN16/32 位

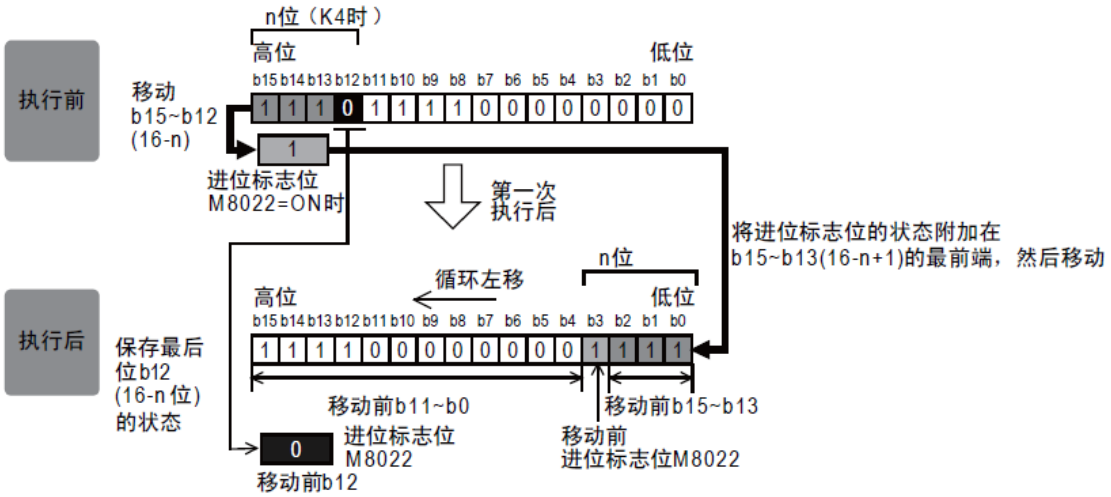
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●	●	●	●	
n							●	●	●	●				

2、功能和动作说明

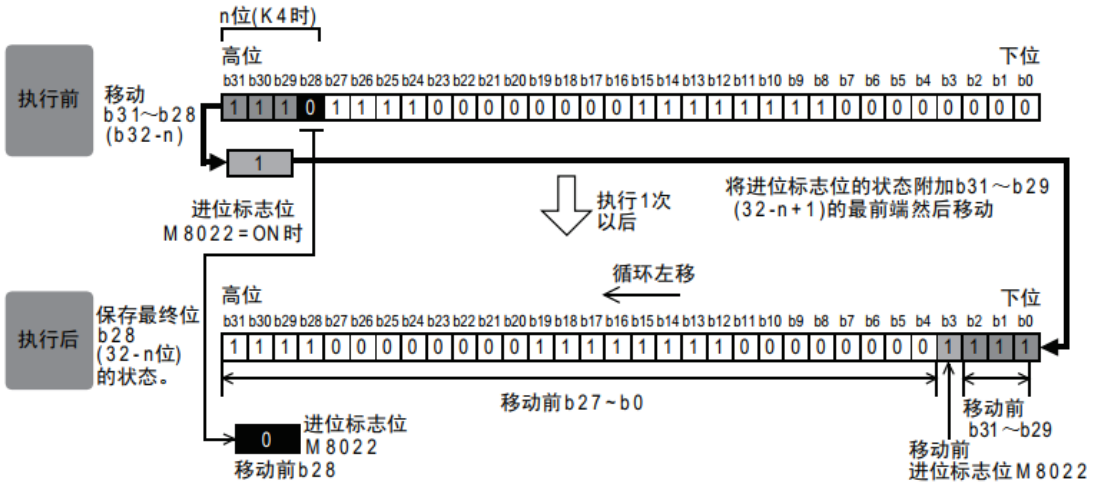
（1）16 位指令

D 的 16 位+1 位(进位标志位 M8022)向左移动 n 位。因为循环回路中有进位标志位，所以如果执行循环移位指令之前 M8022 就先 ON 或 OFF，则会被送入目标操作数中。



（2）32 位指令

将(D+1,D)的 32 位+1 位(进位标志位 M8022)向左移 n 位。最后的位保存在进位标志位(M8022)中。位数指定软元件的情况下，仅 K8(32 位指令)有效。



3、程序举例

参考功能和动作说明。

4、注意事项

（1）相关软元件：M8022，进位标志，最后从最高位移出的位为 1 时置 ON。

（2)连续执行型(RCL、DRCL)指令的场合请注意每个扫描周期(运算周期)都会执行循环移位。

（3)在 D 中指定位数指定软元件时，仅 K4(16 位指令)或 K8(32 位指令)有效。(例如 K4Y010、K8M0)。

3.5.5 SFTR 指令(位右移)

使指定位长度的位软元件每次右移指定的位长度的指令。移动后，从最高位开始传送 n2 点长度的 S 位软元件。

1、指令格式

16bit 9 步		32bit		指令格式
SFTR	SFTRP	\	\	SFTR S D n1 n2

操作数的数据类型如下表

操作数	内容	类型
S	右移后在移位数据中保存的起始位软元件编号	位
D	右移的起始位软元件编号	位

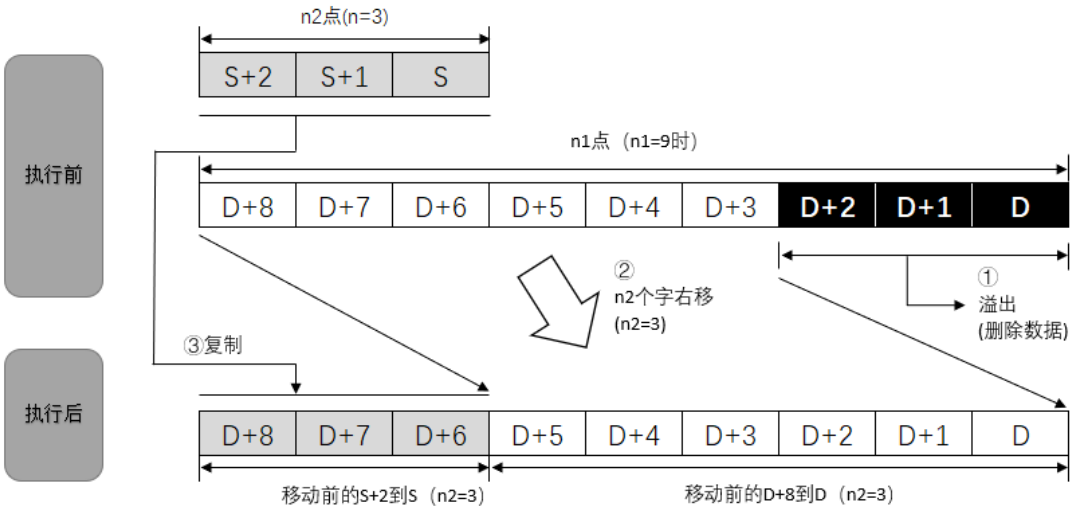
n1	移位数据的位数据长度 n2≦n1≦1024	BIN16
n2	右移的位点数 n2≦n1≦1024	BIN16

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S	●	●	●			●	●	●	●	●				
D		●	●			●	●	●	●					
n1											●	●		
n2											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S													●	
D													●	
n1														
n2							●	●	●	●				

2、功能和动作说明

对于以 D 起始的 n1 位(移位寄存器的长度)数据，右移 n2 位(下记的①、②)。移位后，将 S 开始 n2 位数据传送到从+n1-n2 开始的 n2 位中。



3、程序举例

参考功能和动作说明。

4、注意事项

每次当指令输入从 OFF 变为 ON 时，执行 n2 位移位，每个扫描周期(运算周期)都执行移位。

3.5.6 SFTL 指令(位左移)

使指定位长度的位软元件每次左移指定的位长度的指令。移动后，从最高位开始传送 n2 点长度的 S 位软元件。

1、指令格式

16bit 9 步		32bit		指令格式
SFTL	SFTLP	\	\	SFTL S D n1 n2

操作数的数据类型如下表

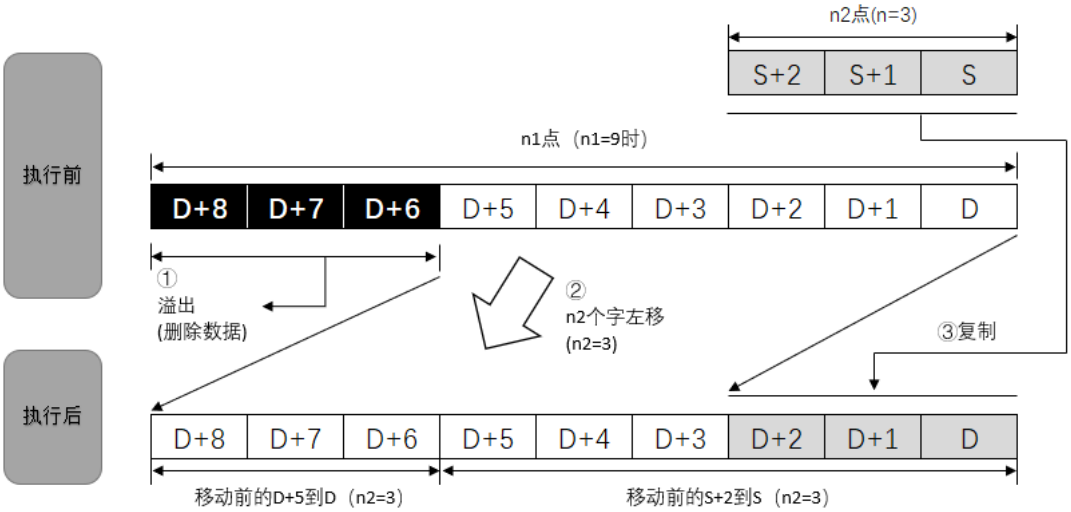
操作数	内容	类型
S	左移后在移位数据中保存的起始位软元件编号	位
D	左移的起始位软元件编号	位
n1	移位数据的位数据长度 n2≦n1≦1024	BIN16
n2	左移的位点数 n2≦n1≦1024	BIN16

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S	●	●	●			●	●	●	●	●				
D		●	●			●	●	●	●					
n1											●	●		
n2											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S													●	
D													●	
n1														
n2							●	●	●	●				

2、功能和动作说明

对于以 D 起始的 n1 位(移位寄存器的长度)数据，左移 n2 位(下记的①、②)。移位后，将 S 开始 n2 位数据传送到从+n1-n2 开始的 n2 位中。



3、程序举例

参考功能和动作说明。

4、注意事项

SFTLP 指令中，每次当指令输入从 OFF 变为 ON 时，执行 n2 位移位，但是请注意 SFTL 指令中，每个扫描周期(运算周期)都执行移位。

3.5.7 WSFR(字右移指令)

将 n1 个字长的字软元件右移 n2 个字的指令

1、指令格式

16bit 9 步		32bit		指令格式
WSFR	WSFRP	\	\	WSFR S D n1 n2

操作数的数据类型如下表

操作数	内容	类型
S	右移后在移位数据中保存的起始软元件编号	BIN16
D	保存右移数据的起始字软元件编号	BIN16

n1	移位数据的字数据长度 2≦n≦512	BIN16
n2	右移的字点数 2≦n≦512	BIN16

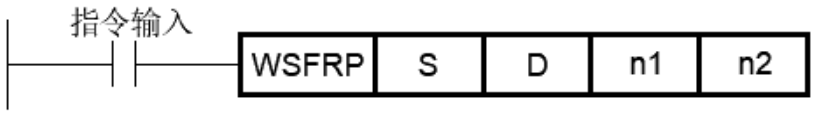
操作数的对象软元件如下表

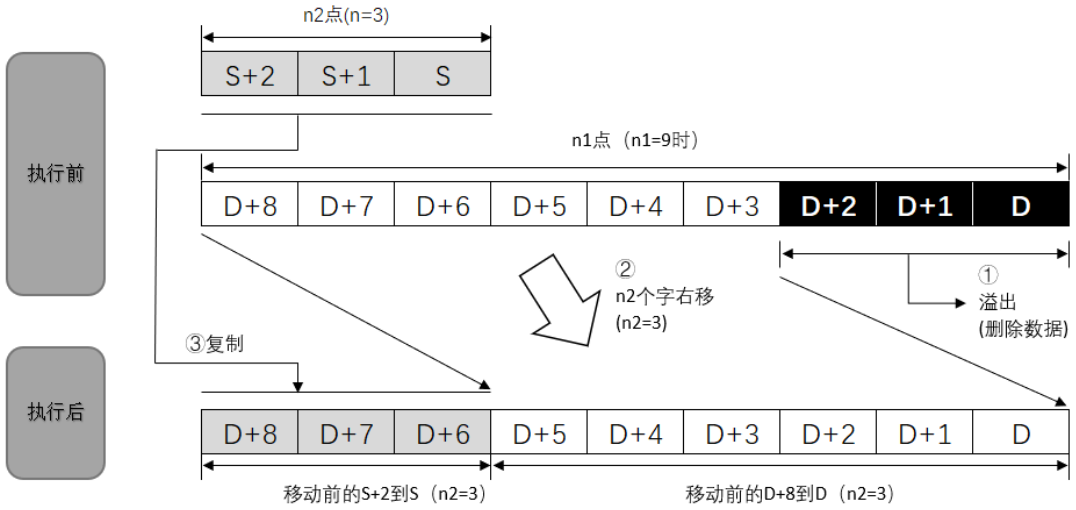
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n1											●	●		
n2											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	
n1														
n2							●	●	●	●				

2、功能和动作说明

对于以 D 起始的 n1 个字软元件，右移 n2 个字（下面的①，②）。

移位后，将 S 开始的 n2 点数据传送（下面的③）到从[D+n1-n2]开始的 n2 点中





3、程序举例

参考功能和动作说明。

4、注意事项

（1）WSFRP 指令中驱动输入 ON 后，移动 n2 个字，但是在 WSFR 指令中每个扫描周期都会执行移动，请务必注意。

（2）传送源 S 和移位软元件 D 重复的时候，发生运算错误。

3.5.8 WSFL(字左移指令)

将字数据信息左移指定字个数的指令。

1、指令格式

16bit 9 步		32bit		指令格式
WSFL	WSFLP	\	\	WSFL S D n1 n2

操作数的数据类型如下表

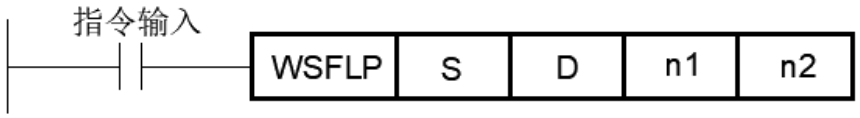
操作数	内容	类型
S	左移后在移位数据中保存的起始软元件编号	BIN16
D	保存左移数据的起始字软元件编号	BIN16
n1	移位数据的字数据长度 2≦n≦512	BIN16
n2	左移的字点数 2≦n≦512	BIN16

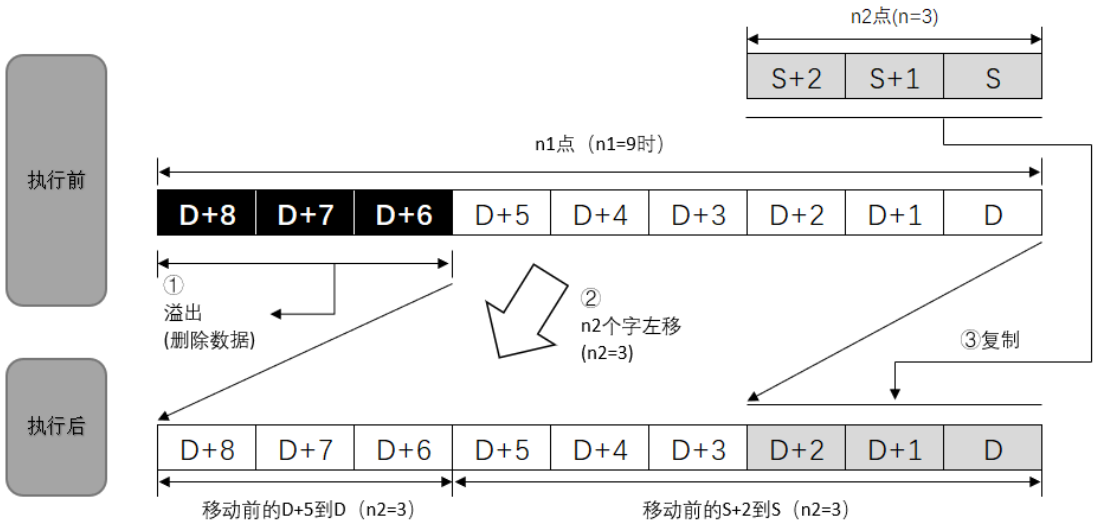
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n1											●	●		
n2											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	
n1														
n2							●	●	●	●				

2、功能和动作说明

将以D 起始的n1 个字软元件，左移n2 个字(下面的①、②)。移位后，将S 开始的n2 点传送(下面的③)到从 D 开始的 n2 点中。





3、程序举例

参考功能和动作说明。

4、注意事项

（1）WSFLP 指令中，每次当指令输入从OFF 变为ON，就执行n2 个字的移位，但是请注意 WSFL 指令中，每个运算周期都执行移位。
（2）传送源 S 和移位软元件 D 重复的时候，发生运算错误。

3.5.9 SFWR(移位写入)

为先入先出和先入后出控制准备的数据写入指令。

1、指令格式

16bit 7 步		32bit		指令格式
SFWR	SFWRP	\	\	SFWR S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存想先入的数据的字软元件编号	BIN16
D	保存数据并移位的起始字软元件编号 2≦n≦512	BIN16
n	保存数据的点数(用于指针时，为+1 后的值)2≦n≦512	BIN16

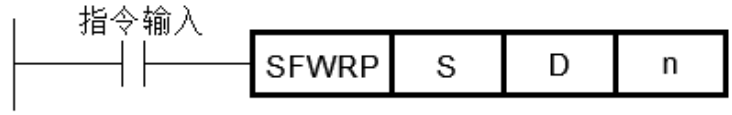
操作数的对象软元件如下表

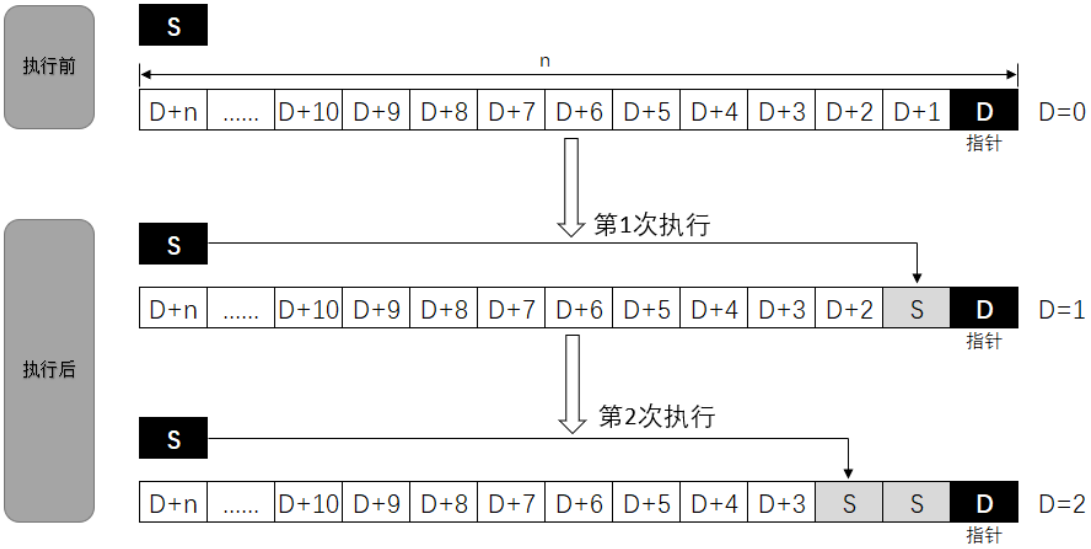
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	
n														

2、功能和动作说明

在 D+1 开始的 n-1 点中依次写入 S 的内容，并对 D 中保存的数据数+1。

例如：D=0 时，S 写入到 D+1 中，写入后 D=1，D=1 时，S 写入到 D+2 中，写入后 D=2。





（1）第一次输入从 OFF 变为 ON 时，S 的内容被保存到 D+1 中，D+1 的内容变为 S 的值。

（2）S 的内容变化后第二次输入从 OFF 变为 ON 时，S 的内容被保存到 D+2 中，D+2 的内容变为 S 的值(由于用连续执行型指令 SFWR，每个扫描周期都依次被保存，因此请用脉冲执行型指令 SFWRP 编程)。

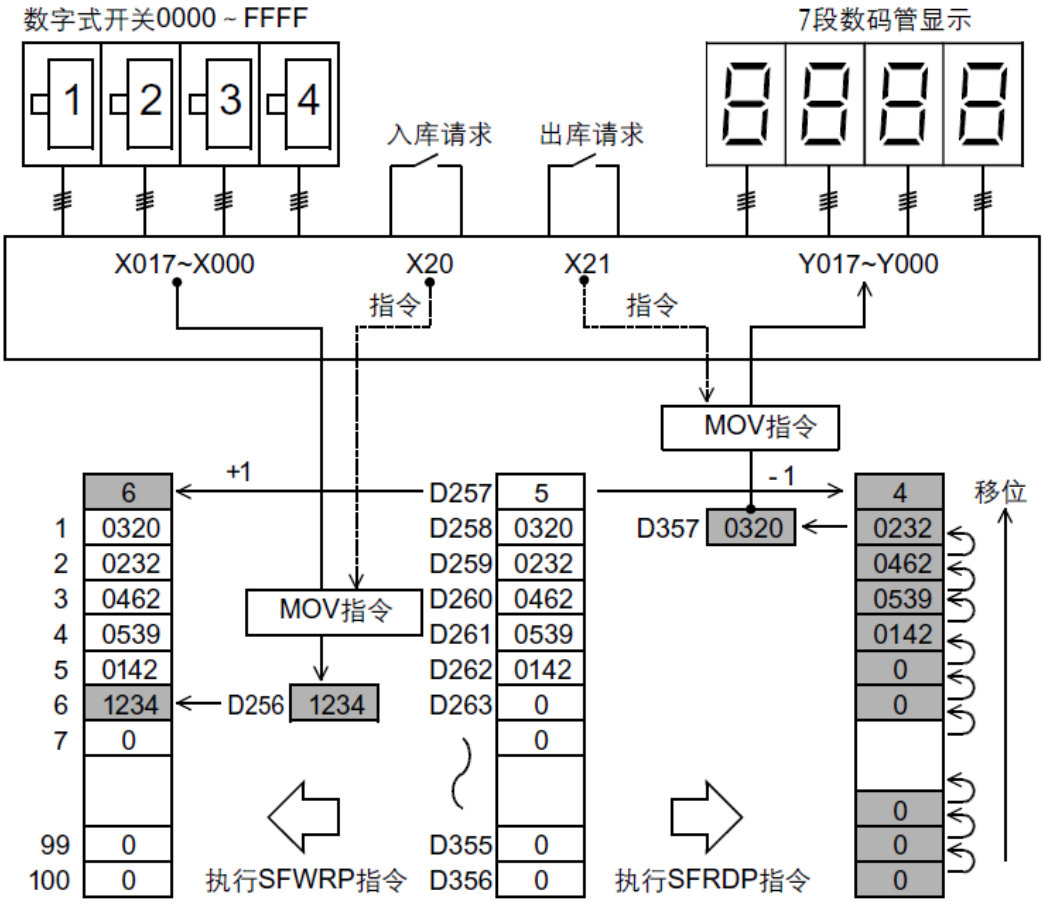
相关软元件

软元件	名称	内容
M8022	进位	指针 D 的内容超过 n-1 时，变为无处理(不写入)，进位标志 M8022 置 ON

相关指令

指令	内容
SFRD	移位读出（先入先出控制用）
POP	读取后入的数据（先入后出控制用）

3、程序举例



4、注意事项

传送源 S 和移位软元件 D 重复的时候，发生运算错误。

3.5.10 SFRD(移位读出指令)

为先入先出控制准备的数据读出指令。

1、指令格式

16bit 7 步		32bit		指令格式
SFRD	SFRDP	\	\	SFRD S D n

操作数的数据类型如下表

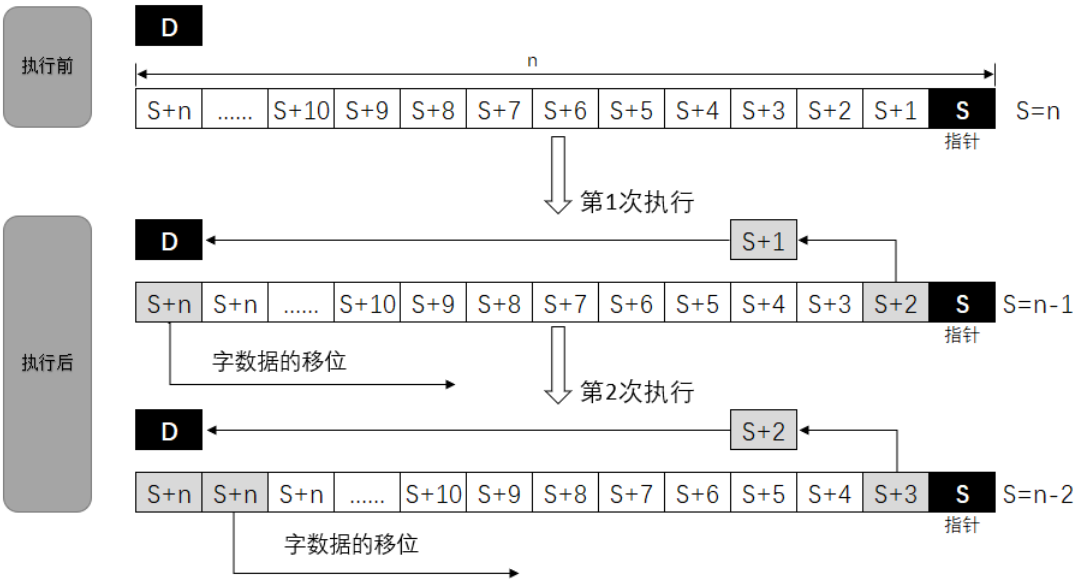
操作数	内容	类型
S	保存数据的起始字软元件编号（最前端为指针，数据从 S+1 开始）	BIN16
D	保存先出的数据的字软元件编号 2≦n≦512	BIN16
n	指定被保存的数据的点数+1（+1 为指针部分）的值 2≦n≦512	BIN16

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S		●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●	●	●	●	
n														

2、功能和动作说明

使用 SFWR 指令被依次写入的 S+1 传送（读出）到 D 中后，从 S+1 开始的 n-1 点逐字右移。S 中保存的数据数 -1。



（1）第一次输入从 OFF 变为 ON 时，S+1 的内容传送（读出）到 D 中。

（2）与此同时，指针 S 的内容-1，左侧的数据逐字右移。（由于用连续执行型指令 SFRD，每个扫描周期都依次被保存，因此请用脉冲执行型指令 SFRDP 编程）

相关软元件

软元件	名称	内容
M8020	零	数据的读出，通常从 S+1 开始执行，但是指针 S 的内容为 0 时，零标志位 M8020 动作[D 为 0 时为无处理，且 M8020 置 ON]

相关指令

指令	内容
SFWR	移位写入（先入先出/先入后出控制用）
POP	读取后入的数据（先入后出控制用）

3、程序举例

参考 SFWR 的程序举例。

4、注意事项

(1)执行读出后的数据

S+n 的内容读出后变为 0。

(2)连续执行型（SFRD）指令的场合

注意每个扫描周期都会一次读出，但 S+n 的内容变为 0。

3.5.11 SFWR2(移位写入指令)

为先入先出和先入后出控制准备的数据写入指令。

1、指令格式

16bit 7 步		32bit		指令格式
SFWR2	\	\	\	SFWR2 S D n1 n2

操作数的数据类型如下表

操作数	内容	类型
S	保存想先入的数据的字软元件编号	BIN16
D	保存数据并移位的起始字软元件编号 2≦n≦512	BIN16
n1	保存数据的点数(用于指针时，为+1 后的值)2≦n≦512	BIN16
n2	数据宽度	BIN16

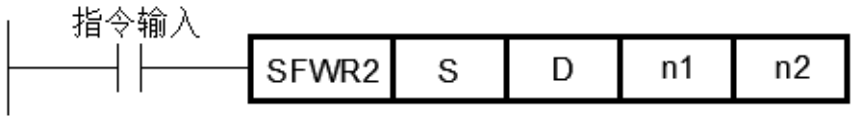
操作数的对象软元件如下表

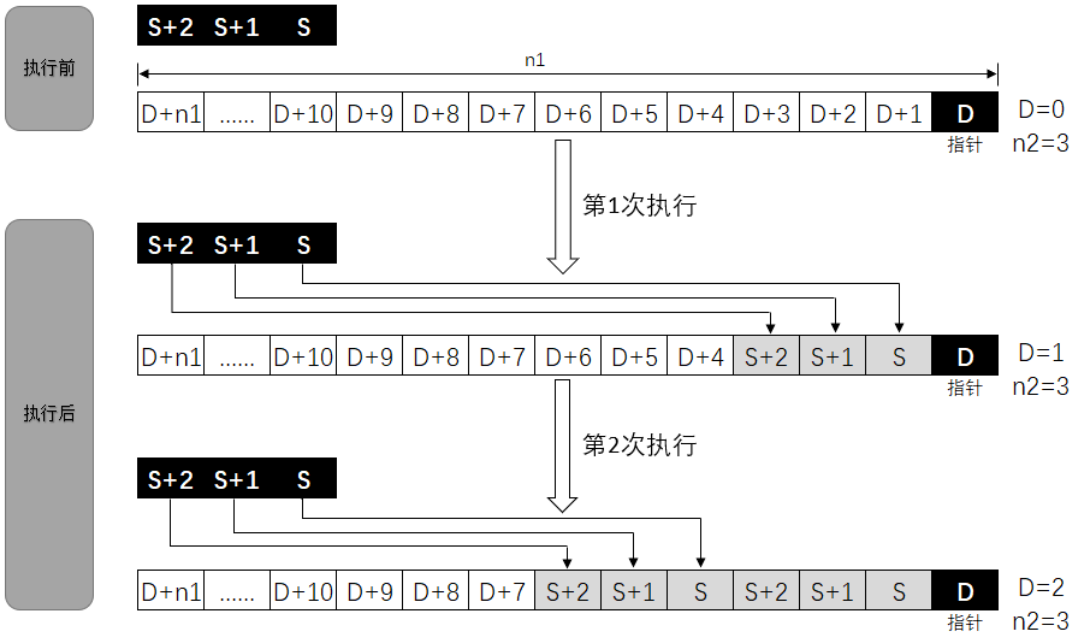
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n1											●	●		
n2											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	
n1														
n2					●	●	●	●	●	●	●	●	●	

2、功能和动作说明

在 D+1 开始的 n1-1 点中依次写入 S 到 S+n2-1 的内容，并对 D 中保存的数据数+1。

例如：D=0，n2=3 时，S 写入到 D+1 中，S+1 写入到 D+2 中，S+2 写入到 D+3 中， D=1，n2=3 时，S 写入到 D+4 中，S+1 写入到 D+5 中，S+2 写入到 D+6 中。





（1）第一次输入从 OFF 变为 ON 时，S 的内容被保存到 D+1 中，S+1 的内容被保存到 D+2 中，S+2 的内容被保存到 D+3 中，D+1 的内容变为 S 的值，D+2 的内容变为 S+1 的值，D+3 的内容变为 S+2 的值。

（2）S，S+1，S+2 的内容变化后第二次输入从 OFF 变为 ON 时，S 的内容被保存到 D+4 中，S+1 的内容被保存到 D+5 中，S+2 的内容被保存到 D+6 中，D+4 的内容变为 S 的值，D+5 的内容变为 S+1 的值，D+6 的内容变为 S+2 的值。（由于用连续执行型指令 SFWR2，每个扫描周期都依次被保存，因此使能信号请使用脉冲型）。

相关指令

指令	内容
SFRD2	移位读出（先入先出控制用）
POP2	读取后入的数据（先入后出控制用）

3、程序举例

参考功能和动作说明

4、注意事项

传送源 S 和移位软元件 D 重复的时候，发生运算错误。

3.5.12 SFRD2(移位读出指令)

为先入先出控制准备的数据读出指令。

1、指令格式

16bit 7 步		32bit		指令格式
SFRD2	\	\	\	SFRD2 S D n1 n2

操作数的数据类型如下表

操作数	内容	类型
S	保存数据的起始字软元件编号（最前端为指针，数据从 S+1 开始）	BIN16
D	保存先出的数据的字软元件编号 2≦n≦512	BIN16
n 2	指定被保存的数据的点数+1（+1 为指针部分）的值 2≦n≦512	BIN16
n 1	数据宽度	BIN16

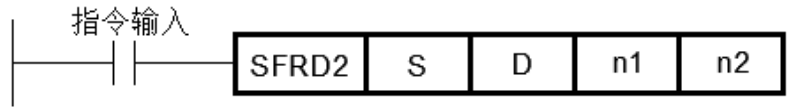
操作数的对象软元件如下表

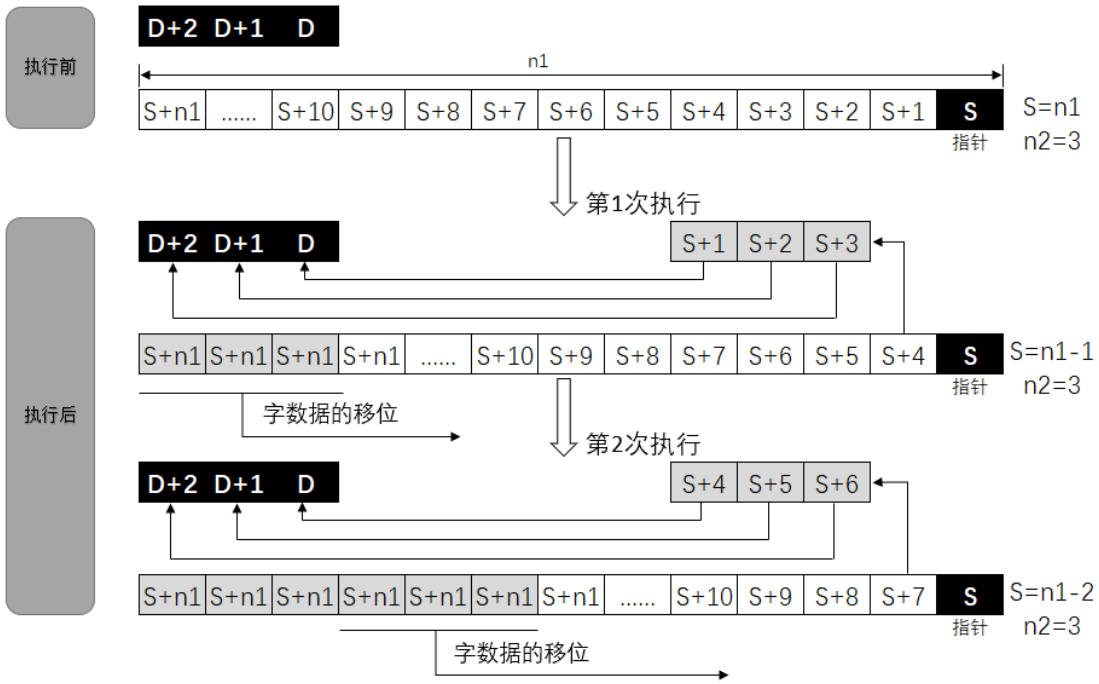
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n1											●	●		
n2											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S		●	●	●	●	●	●	●	●	●			●	

D		●	●	●	●	●	●	●	●	●	●	●	●	
n1														
n2					●	●	●	●	●	●	●	●	●	

2、功能和动作说明

使用 SFWR2 指令被依次写入的 S+1 到 S+n1 传送（读出）到 D 到 D+n2-1 中后，从 S+1 开始的 n-1 点逐字右移 n2 个字。S 中保存的数据数 -1。





（1）第一次输入从 OFF 变为 ON 时，S+1 的内容传送（读出）到 D 中，S+2 的内容传送（读出）到 D+1 中，S+3 的内容传送（读出）到 D+2 中。

（2）与此同时，指针 S 的内容-1，左侧的数据逐字右移 n2 个字。（由于用连续执行型指令 SFRD，每个扫描周期都依次被保存，因此请用脉冲执行型指令 SFRDP 编程）

相关指令

指令	内容

SFWR2	移位写入（先入先出/先入后出控制用）
POP2	读取后入的数据（先入后出控制用）

3、程序举例

参考功能和动作说明。

4、注意事项

(1)执行读出后的数据

S+n 的内容读出后变为 0。

(2)连续执行型（SFRD）指令的场合

注意每个扫描周期都会一次读出，但 S+n 的内容变为 0。

3.6 数据处理指令

3.6.1 ZRST 指令(成批复位)

2 个指定的软元件之间执行成批复位的指令。用于在中断运行后从初期开始运行时，以及对控制数据进行复位时。

1、指令格式

16bit 5 步		32bit		指令格式
ZRST	ZRSTP	\	\	ZRST D1 D2

操作数的数据类型如下表

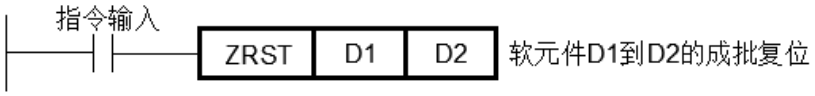
操作数	内容	类型
D1	成批复位的最前端的位/字软元件编号	BIN16 位
D2	成批复位的末尾的位/字软元件编号	BIN16 位

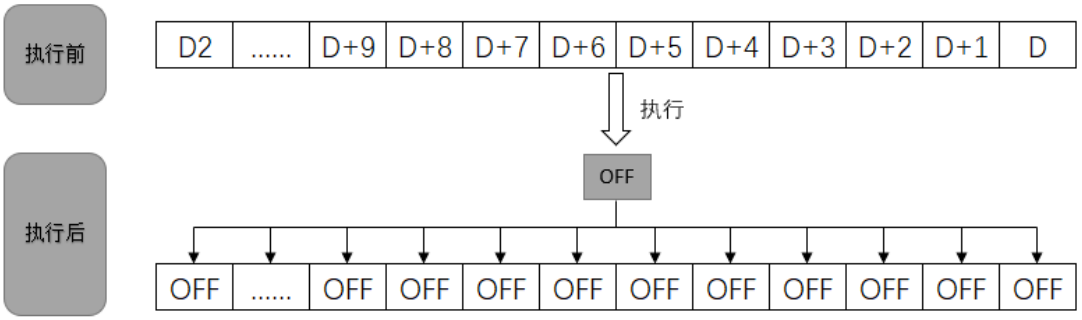
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D1		●	●			●	●	●	●					

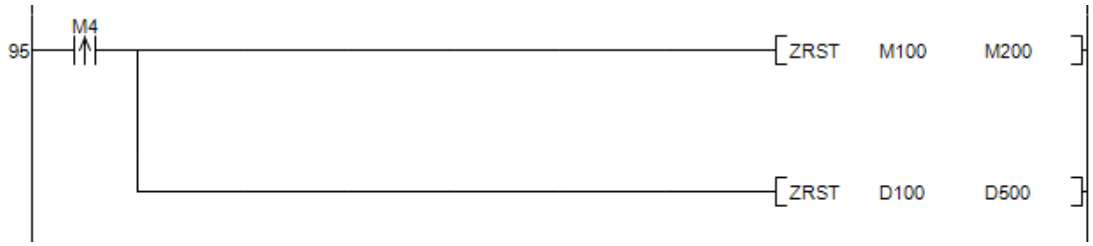
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D1					●	●	●	●	●	●			●	
D2					●	●	●	●	●	●			●	

2、功能和动作说明





3、程序举例



4、注意事项

（1）D1，D2 必须指定为同一种类的软元件，且编号 D1≦D2 编号，当 D1 编号大于 D2 时,D1 中指定的软元件只复位 1 点。

（2） ZRST 作为 16 位处理指令。也可以在 D1，D2 中指定 32 位计数器，但是不能出现 D1 和 D2 的寄存器位数不一样的情况。

3.6.2 ZSET 指令(成批置位)

2 个指定的软元件之间执行成批置位的指令。用于在中断运行后从初期开始运行时，以及对控制数据进行置位时。

1、指令格式

16bit 5 步		32bit		指令格式
ZSET	ZSETP	\	\	ZSET D1 D2

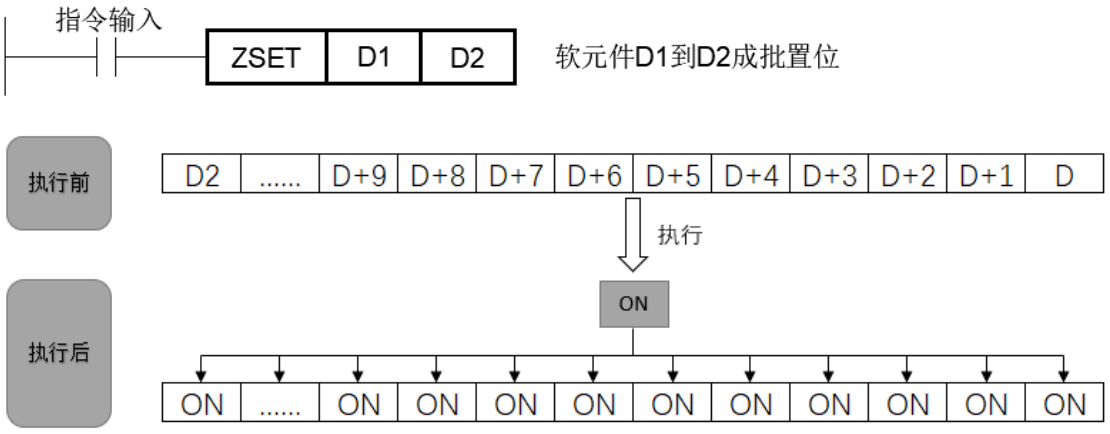
操作数的数据类型如下表

操作数	内容	类型
D1	成批置位的最前端的位/字软元件编号	BIN16 位
D2	成批置位的末尾的位/字软元件编号	BIN16 位

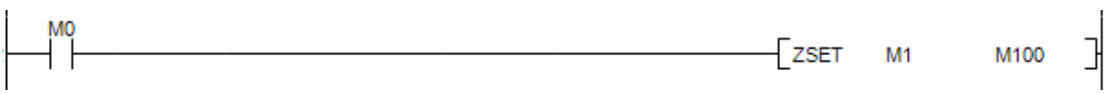
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D1		●	●			●	●	●	●					
D2		●	●			●	●	●	●					
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D1													●	
D2													●	

2、功能和动作说明



3、程序举例



4、注意事项

（1）D1，D2 必须指定为同一种类的软元件，且编号 D1≦D2 编号，当 D1 编号大于 D2 时,D1 中指定的软元件只复位 1 点。

3.6.3 DECO 指令(译码)

将数字数据中任意一个转换成 1 点的 ON 位的指令。根据 ON 位的位置可以将位编号读成数值。

1、指令格式

16bit 7 步		32bit		指令格式
DECO	DECOP	\	\	DECO S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存要译码的数据，或是数据的字软元件编号	BIN16 位
D	保存译码结果的位/字软元件编号	BIN16 位
n	保存译码结果的软元件的位点数(n=1～8)(n=0 时为不处理)	BIN16 位

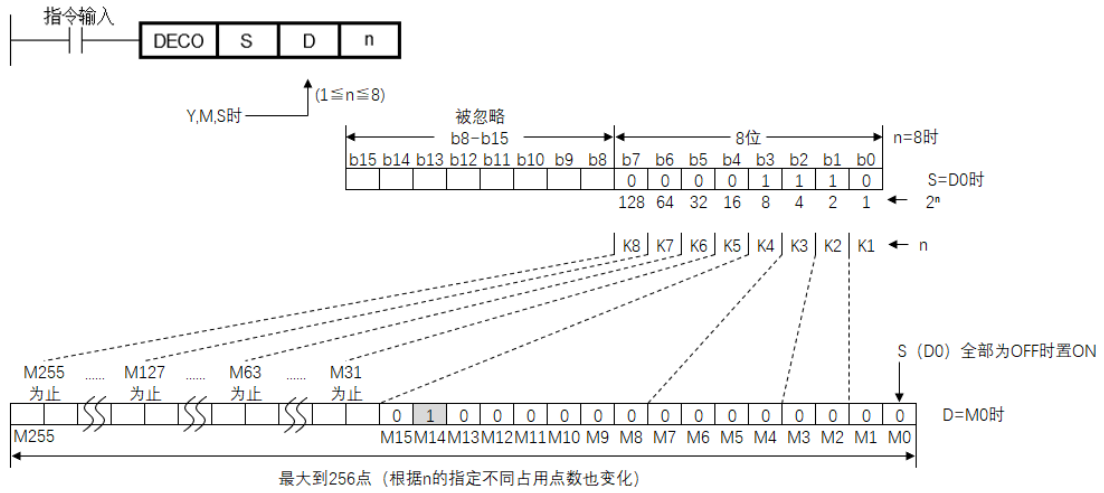
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S	●	●	●			●	●	●	●		●	●		
D		●	●			●	●	●	●					
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●	●	●	●	
D					●	●	●	●	●	●			●	
n														

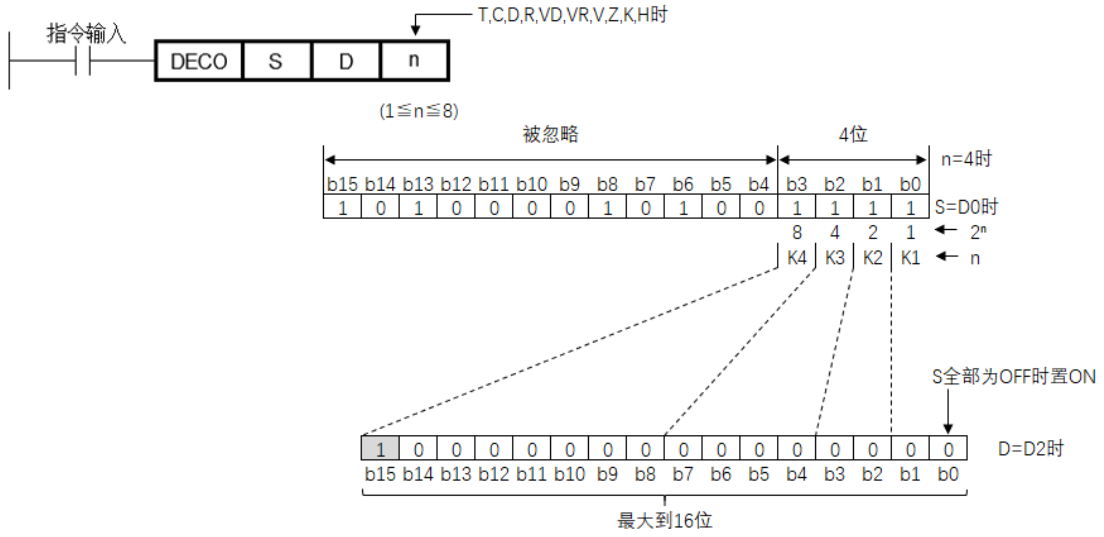
2、功能和动作说明

S 的值相对应的 D+2^n-1 中的 1 个置 ON。

（1）D 为位软元件时：1≦n≦8，S 中指定的软元件的 n 位数(1≦n≦8)，在 D 中被译码。S 都为 0 时，位软元件 D 为 ON。n=8 时， 最大 2^8=256 点。



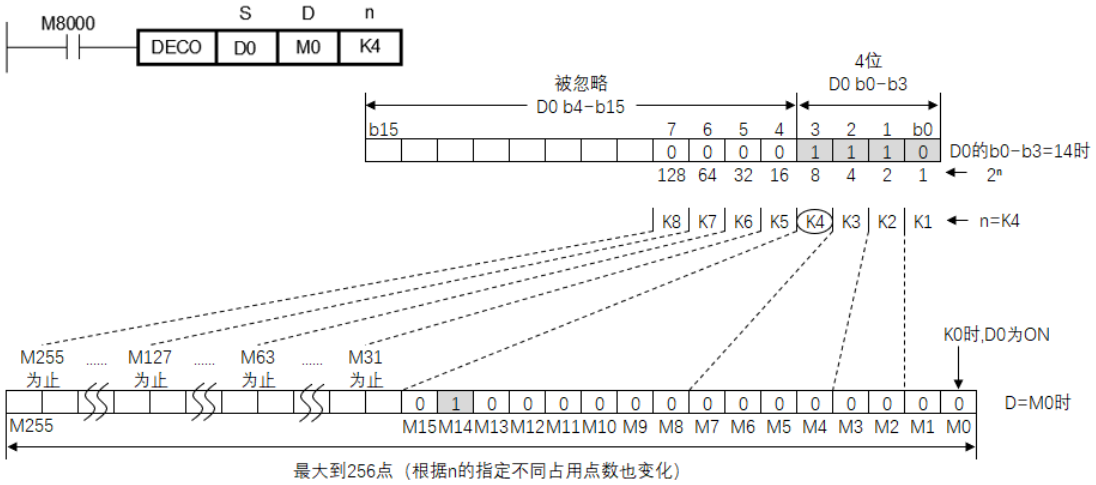
（2）D 为字软元件时：1≦n≦4，S 的低 n 位在 D 中被译码。S 都为 0 时，位软元件 D 的 b0 为 ON。n≦3 时 D 的高字节都为 0。



3、程序举例

（1）D 为位软元件

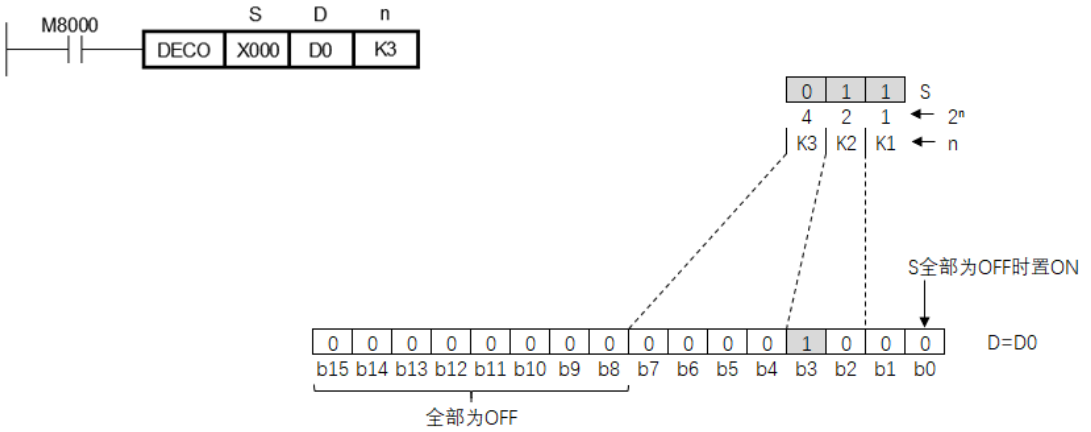
D0 的值(当前值取 14)在 M0～M15 中译码。



- D0 的 b0～b3 的值为 14(0+2+4+8)时，M0 开始的第 15 号的 M14 为 1(ON)。
- D0=0 时，M0 为 1(ON)。
- 使 n=K4，根据 D0(0～15)的数值，M0～M15 中任意一个为 1 点(ON)。
- 若使 n 在 K1～K8 之间变化，D0 就可以对应 0～255 的数值。

（2）D 为字软元件

X000～X002 的值(X000、X001 为 ON，X002 为 OFF)在 D0 中译码。



- X000～X002 的值为 3(1+2+0)时，b0 开始的第 4 号的 b3 为 1(ON)。
- X000～X002 都为 0(OFF)时，b0 为 1(ON)。

4、注意事项

（1）n 在 K1～K8 之间变化时，注意目的寄存器 D 为位软元件的范围不能与其他控制重复。

（2）n 为 0 时，指令不处理。

3.6.4 ENCO 指令(编码)

求出在数据中 ON 位的位置的指令。

1、指令格式

指令格式见下表

16bit 7 步		32bit		指令格式
ENCO	ENCOP	\	\	ENCO S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存要编码的数据，或是数据的字软元件编号	BIN16 位
D	保存编码结果的字软元件编号	BIN16 位
n	保存译码结果的软元件的位点数(n=1～8)(n=0 时为不处理)	BIN16 位

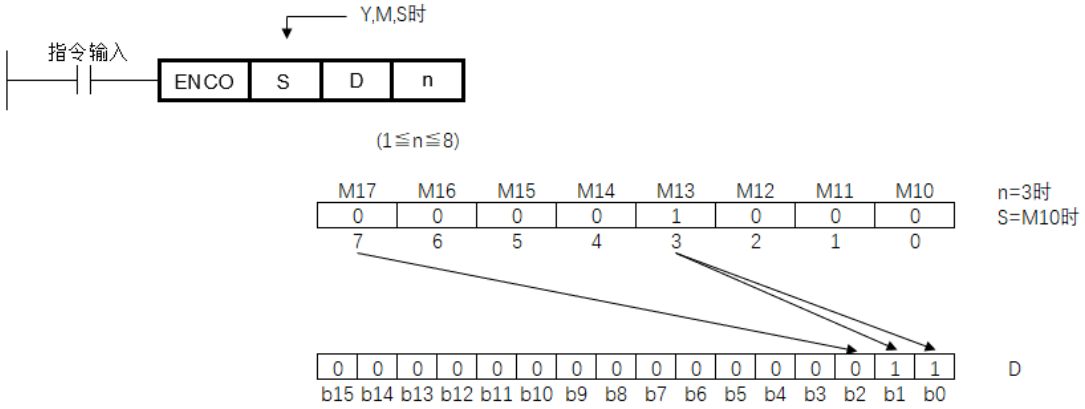
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S	●	●	●			●	●	●	●					
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●	●	●	●	
D					●	●	●	●	●	●	●	●	●	
n														

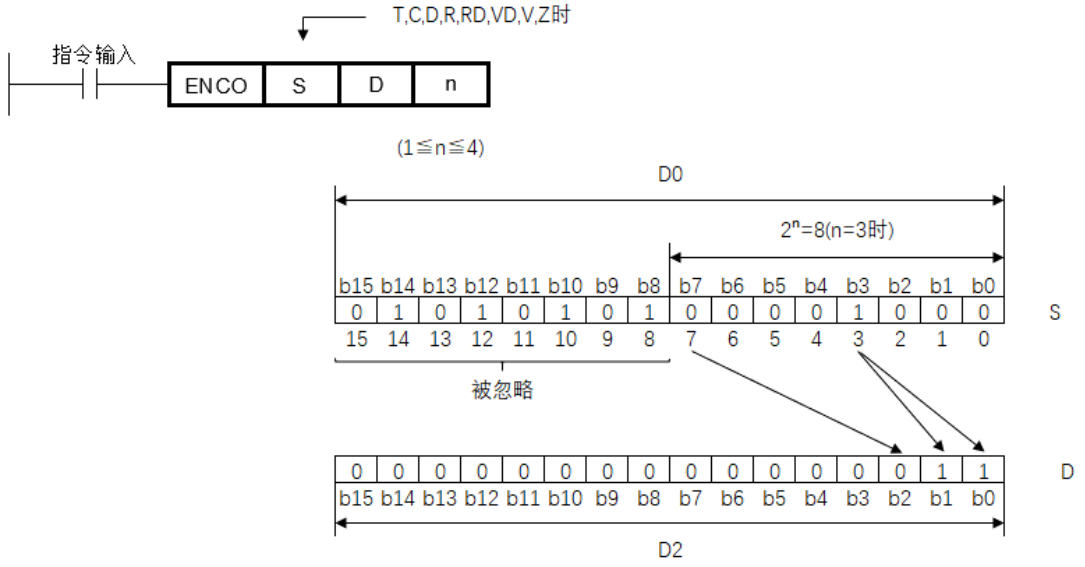
2、功能和动作说明

在 D 中保存 S 的 2^n 位编码后的值，编码就是将 ON 位的位置转换成 BIN 数据。

（1）D 为位软元件：1≦n≦8，S 开始 2^n 个的 ON 位位置，在 D 中编码。n=8 时，S 为最 256 点。D 的编码结果从高位到低位 n 位，全部为 0(OFF)。



（2）D 为字软元件：1≦n≦4，到 S 中指定的软元件的位 2^n 个为止的 ON 位位置，都在 D 中编码。D 的编码结果从高位到低位 n 位，全部为 0(OFF)。



3、程序举例

参考功能和动作说明。

4、注意事项

S 的数据中多个位为 ON 的情况下，忽略低位侧，对高位侧的 ON 位进行编码。

3.6.5 SUM 指令(ON 位数)

计算在指定的软元件的数据中有多少个为“1”(ON)的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
SUM	SUMP	DSUM	DSUMP	SUM S D

操作数的数据类型如下表

操作数	内容	类型
S	保存数据的字软元件编号	BIN16/32 位
D	保存结果数据的字软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

（1）16 位指令

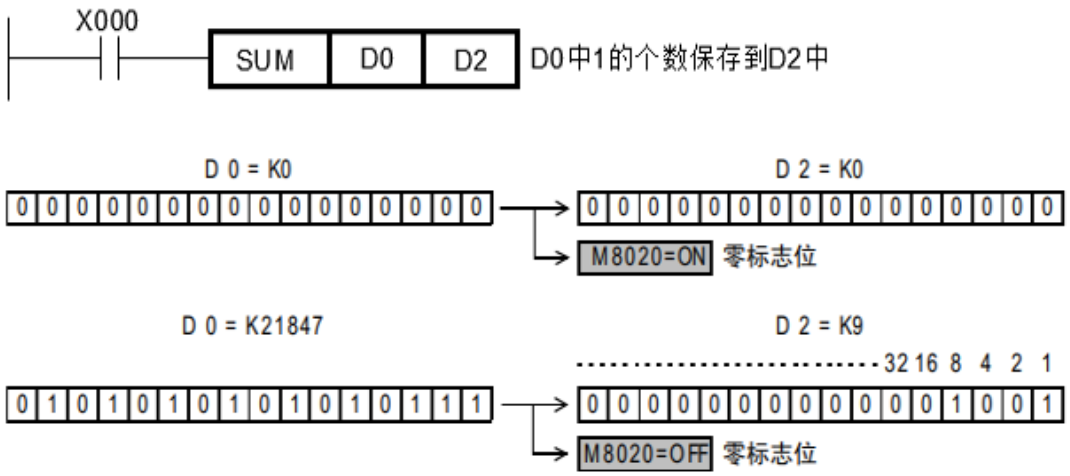
对 S 中为 ON 的位进行计数后保存到 D 中。S 都为 0（OFF 时），零位标志位 M8029 为 ON。

（2）32 位指令

对（S+1,S ）中为 ON 的位进行计数后保存到 D 中。D 中保存 ON 为的点数，D+1 中保存 KO。(S+1,S)都为 0（OFF 时），零位标志位 M8029 为 ON。

3、程序举例

X000 为 ON 时，对 D0 中 ON 的位数计数后保存到 D2 中



4、注意事项

相关标志位 M8020 的动作。

3.6.6 BON 指令(ON 位的判断)

检查软元件中指定位的位置为 ON 还是 OFF 的指令

1、指令格式

16bit 7 步		32bit 13 步		指令格式
BON	BONP	DBON	DBONP	BON S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存数据的字软元件编号	BIN16/32 位
D	驱动的位软元件编号	位
n	要判定的位置[n: 0～15(16 位指令)，n: 0～31(32 位指令)]	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件	常数	实数	字符串

	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
S											●	●		
D		●	●			●	●	●	●	●				
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D													●	
n							●	●	●	●				

2、功能和动作说明

（1）16 位指令

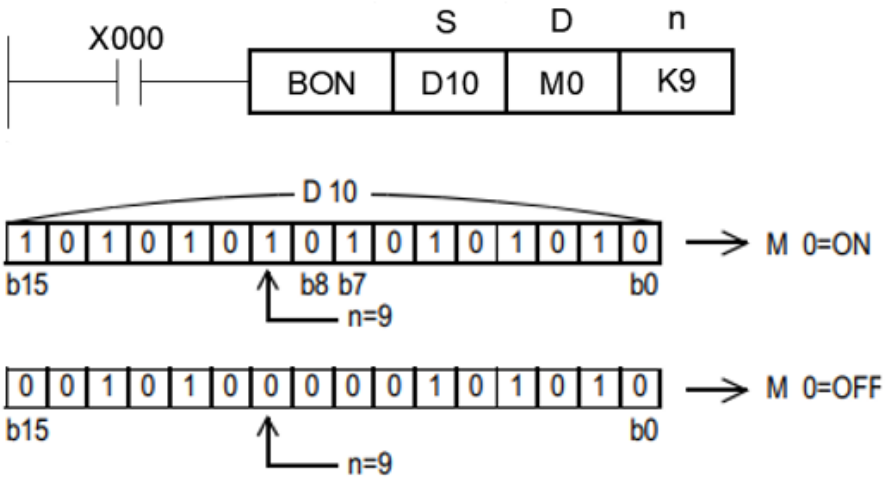
S 的 n 位的状态（ON/OFF）输出到 D。

（2）32 位指令

（S+1，S）中的 n 位的状态（ON/OFF）输出到 D。

3、程序举例

D10 中的 n=第 9 位为 1(ON)时，M0 为 1(ON)。



4、注意事项

将 D、R 指定为 32 位指令的 n 时，[n+1,n]的 32 位值便生效，敬请注意。
DBON D0 M0 R0 时，则 n=[R1, R0]

3.6.7 MEAN 指令(平均值)

求数据的平均值的指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
MEAN	MEANP	DMEAN	MEANP	MEAN S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存想要的平均值数据的起始字软元件编号	BIN16/32 位
D	保存取得的平均值数据的字软元件编号	BIN16/32 位
n	平均数据数(n=1～64)	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●	●	●	●	
n							●	●	●	●				

2、功能和动作说明

（1）16 位指令

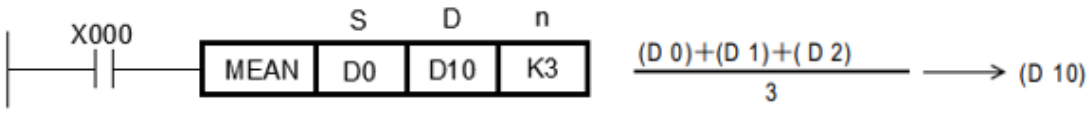
将 S 开始的 n 个 16 位数据的平均值保存到 D 中，余数舍去。

（2）32 位指令

将（S+1，S）开始的 n 个 32 位数据的平均值保存到（D+1,D）中，余数舍去。

3、程序举例

将 D0、D1、D2 的数据相加，除以 3 后求得的值保存到 D10 中



4、注意事项

（1）软元件编号溢出时，在可能的范围内将 n 作为较小的值处理。

（2）注意 n 的范围(n=1～64)，超出报错。

3.6.8 ANS 指令(信号报警器置位)

对信号报警器用的状态(S900～S999)进行置位用的指令。

1、指令格式

16bit 7 步		32bit		指令格式
ANS	ANSP	\	\	ANS S m D

操作数的数据类型如下表

操作数	内容	类型
S	判断时间的计时定时器编号	BIN16/32 位
m	判断时间的数据[m=1～32,767(100ms 单位)]	BIN16/32 位
D	设置的信号报警器软元件	位

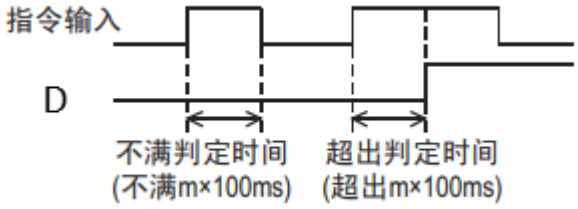
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
m											●	●		
D						●								
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●								●	
m							●	●	●	●				

D												●	

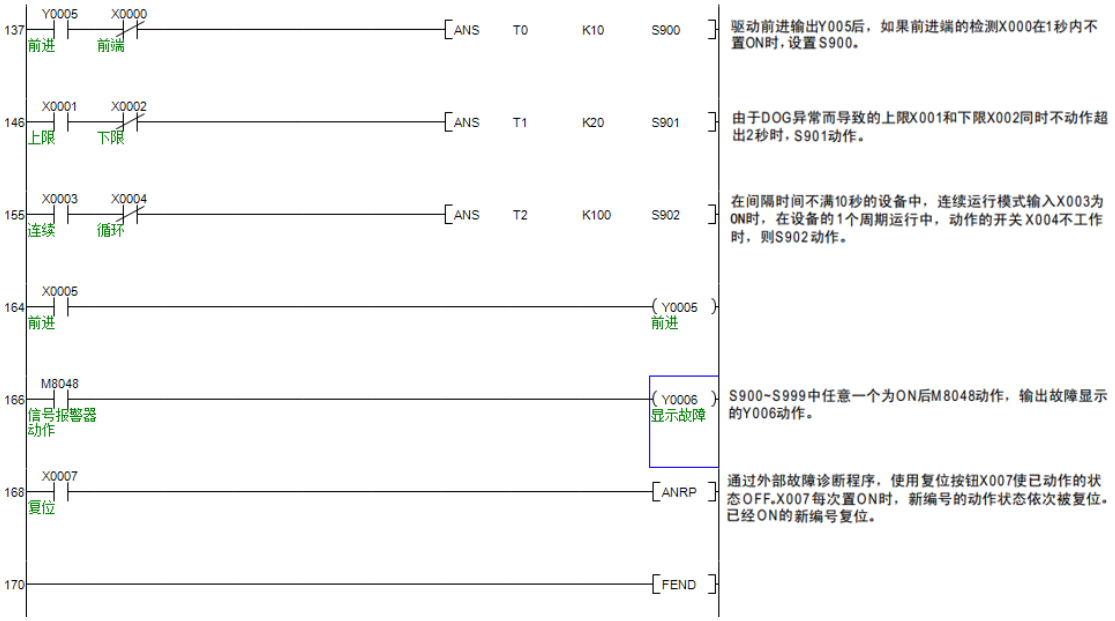
2、功能和动作说明

16 位运算指令，指令输入超出判定时间[m×100ms，定时器 S]以上持续为 ON 时，D 置位。指令输入在不满判定时间[m×100ms]就已 OFF 的情况下，复位判定用定时器 S 的当前值,不置位 D。此外，指令输入 OFF 后，判定用定时器复位。



3、程序举例

如下所示，编写诊断外部故障用的程序，如监控 D8049(ON 状态最小编号)的内容时，会显示 S900～S999 中为 ON 的状态的最小编号。同时发生多个故障时，排除了最小编号的故障后可以得知下一个故障编号 。



4、注意事项

（1）S 为 T 软元件时，范围为 T00～T199，报警用状态软元件范围是 S900～S999。

（2）注意与 S900～S999 相关的特殊寄存器的动作。

软元件	名称	内容
M8048	信号报警器动作	状态 S900～S999 中任一动作的时候，M8048 置 ON。

3.6.9 ANR 指令(信号报警器复位)

对信号报警器(S900～S999)中已经置 ON 的小编号进行复位。

1、指令格式

16bit 1 步		32bit		指令格式
ANR	ANRP	\	\	ANR

2、功能和动作说明

16 位运算指令，ANR 指令输入为 ON 后，将信号报警器 S900～S999 中运行中的状态复位。如有多个状态动作时，复位编号最小的一个状态。再次使指令输入为 ON 后，在动作的信号报警器用状态(S900～S999)中，下一个最小的编号被复位。

软元件	名称	内容
M8048	信号报警器动作	状态 S900～S999 中任一动作的时候，M8048 置 ON。

3、程序举例

可参考 ANS 指令。

4、注意事项

（1）相关寄存器的动作:

软元件	名称	内容
M8048	信号报警器动作	状态 S900～S999 中任一动作的时候，M8048 置 ON。

（2）使用 ANR 指令时，每个运算周期都依次复位最小标号，ANPR 指令只执行 1 个运算周期，根据实际需求来选择对应的指令以复位对应的报警器。

3.6.10 SQR 指令(BIN 开放运算)

求平方根(开根号)的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
SQR	SQRP	DSQR	DSQRP	SQR S D

操作数的数据类型如下表

操作数	内容	类型
S	保存要被开平方根运算数据的字软元件编号	BIN16/32 位
D	保存被执行了开平方根运算数据的数据寄存器编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

（1）16 位指令

计算出 S 的数据的平方根后，保存到 D 中。

（2）32 位指令

计算出（S+1，S）的数据的平方根后，保存到（D+1，D）中。

3、程序举例

4、注意事项

（1）当运算结果为小数时，舍弃小数部分，将整数结果填入 D，M8021 置 ON。

（2）结果为 0 时，M8020 置 ON。

3.6.11 FLT 指令(BIN 整数到 2 进制浮点数转换)

将 BIN 整数值转换成 2 进制浮点数(实数)的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
FTL	FTLP	DFTL	DFTLP	FTL S D

操作数的数据类型如下表

操作数	内容	类型
S	保存 BIN 整数值的数据寄存器编号	BIN16/32 位
D	保存 2 进制浮点数(实数)的数据寄存器编号	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

（1）16 位指令

将 S 的 BIN 整数值数据转换成 2 进制浮点数(实数)值后，保存在（D +1，D）中。

（2）32 位指令

将（S+1，S）的 BIN 整数值数据转换成 2 进制浮点数(实数)值后，保存到（D+1，D）中。

3、程序举例

无

4、注意事项

16 位和 32 位指令的目的操作数都是占用 2 个字软元件。

3.7 方便指令

3.7.1 SER 指令(数据检索)

从数据表中检索相同数据、最大值、最小值的指令。

1、指令格式

16bit 9 步		32bit 17 步		指令格式
SER	SERP	DSER	DSERP	SER S1 S2 D n

操作数的数据类型如下表

操作数	内容	类型
S1	检索相同数据、最大值、最小值的起始软元件编号	BIN16/32 位
S2	检索相同数据、最大值、最小值的参考值或是其保存目标软元件编号	BIN16/32 位
D	检索相同数据、最大值、最小值后，保存这些个数的起始软元件编号	BIN16/32 位
n	检索相同数据、最大值、最小值的个数 [16 位指令时：1-256，32 位指令时：1-128]	BIN16/32 位

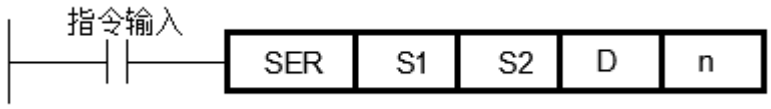
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
S2											●	●		
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●			●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	

D		●	●	●	●	●	●	●	●	●			●	
n							●	●	●	●				

2、功能和动作说明

对以 S 开始的 n 个数据进行检索，检索与 S2 的数据相同的数据，并将结果保存在 D 到 D+4 中。



1） 被检索的数据的内容及结果

a）存在相同数据时

在以 D 开始的 5 个软元件中，保存相同数据的个数、初次/最终的位置以及最大值、最小值的位置。

b）不存在相同数据时

在以 D 开始的 5 个人软元件中，保存相同数据的个数、初次/最终的位置以及最大值、最小值的位置。但是，在以 D 开始的 3 个软元件（相同数据的个数、初次/最终的位置）中，0 被保存。

3、程序举例

无

4、注意事项

（1）大小的比较

以代数方式执行（-10<2）

（2） 存在多个最小值最大值时

分别保存各后侧的位置

（3） 软元件的占用点数

驱动指令后，检索结果 D 会占用如下的软元件点数。

请注意不要与其他控制中使用的软元件重复。

1）16 位运算时

占用[D,D+1,D+2,D+3,D+4]5 点。

2）32 位运算时

占用[D+1,D],[D+3,D+2],[D+5,D+4] ,[D+7,D+6] ,[D+9,D+8]10 点。

3.7.2 TTMR 指令(示教定时器)

测量 TTMR 指令为 ON 的时间的指令。

采用按钮来调节定时器的设定时间的情况下可以使用。

1、指令格式

16bit 5 步		32bit		指令格式
TTMR				TTMR D n

操作数的数据类型如下表

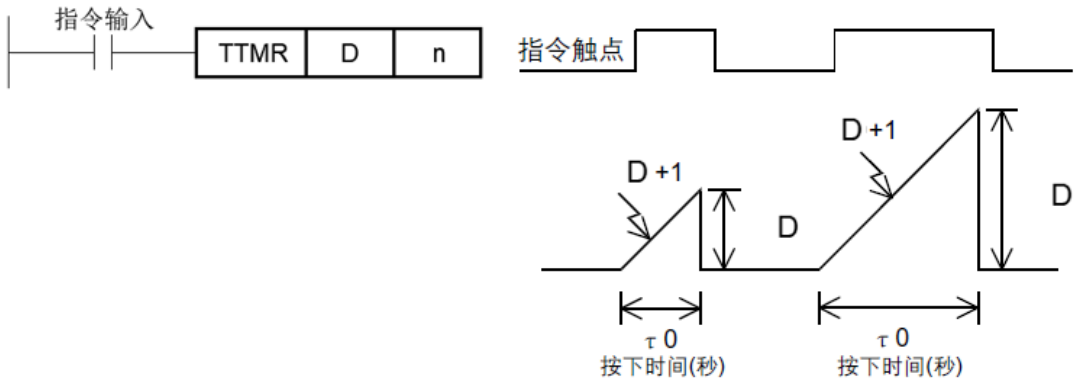
操作数	内容	类型
D	保存示教数据的软元件编号	BIN16 位
n	示教数据乘以的倍率数[K0-K2/H0-H2]	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D							●	●	●	●			●	
n							●	●	●	●				

2、功能和动作说明

以秒为单位，对指令输入（按键）的按下时间进行测量。然后乘以倍率（10^n）后传到 D 中。



被传到D中的时间是这样的，按下时间为τ0（1秒单位）时，根据n的倍率得到实际的D值，如下所示。

n	倍率	D
K0	τ0	D*1
K1	10τ0	D*10
K2	100τ0	D*100

3、程序举例

4、注意事项

（1）当指令触点为 OFF 时

按下时间的当前值[D+1]被复位，示教时间 D 不改变。

（2）软元件的占用点数

以示教时间 D 中指定的软元件为起始，占用 2 点。

请注意不要与其它控制中使用的软元件重复。

D 示教时间

D+1 按下时间的当前值

3.7.3 STMR 指令(特殊定时器)

用于轻松制作断开延时定时器、单脉冲定时器、闪烁定时器的指令。

1、指令格式

16bit 7 步		32bit		指令格式
STMR				STMR S m D

操作数的数据类型如下表

操作数	内容	类型
S	使用的定时器编号[T0-T199(100ms 定时器)]	BIN16 位
m	定时器的设定值[1-32767]	BIN16 位
D	被输出的起始位编号（占用 4 点）	位

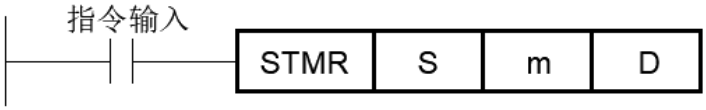
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
m											●	●		
D		●	●			●	●	●	●	▲				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●								●	
m							●	●	●	●				
D													●	

▲：Dx.y 不能变址修饰（V,Z）

2、功能和动作说明

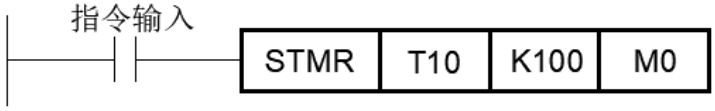
将 m 中指定的值作为 S 中指定的定时器的设定值，从 D 开始输出 4 点。



3、程序举例

断开延时定时器 单脉冲定时器

在 S 中分配 T10，m 中分配 K100，D 中分配 M0

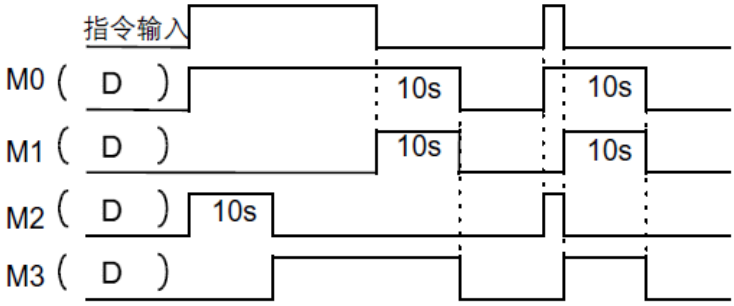


M0[D]:指令触点 OFF 后，经过定时器的设定时间之后，再 OFF 的断开延时定时器。

M1[D+1]:指令触点从 ON 变为 OFF 后为 ON，经过定时器设定的时间后变为 OFF 的单脉冲定时器。

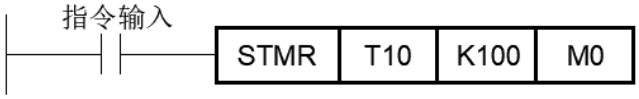
M1[D+2]:占用，用于闪烁。

M1[D+3]:占用。



闪烁

用在 D+3 的 b 触点断开该指令，通过如下所示程序，在 D+1,D+2 中输出闪烁。占用 D，D+3。

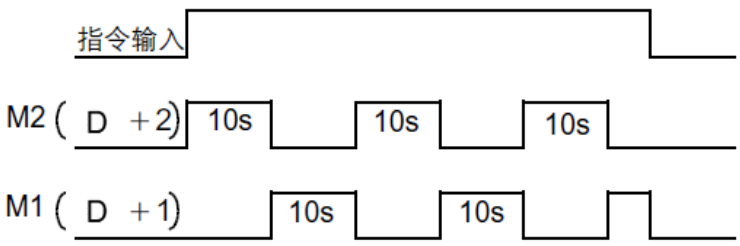


M0[D]:指令触点 OFF 后，经过定时器的设定时间之后，再 OFF 的断开延时定时器。

M1[D+1]:以定时器的间隔时间，重复执行 ON/OFF 的闪烁（a 触点）

M1[D+2]:以定时器的间隔时间，重复执行 ON/OFF 的闪烁（b 触点）

M1[D+3]:占用。



4、注意事项

（1）指定定时器的使用

这个指令中所指定的定时器编号，不能在其他普通回路中（OUT 指令等）重复使用。

（2）软元件的占用点数

以 D 中指定的软元件为起始，占用 4 点。

请注意不要与其它控制中使用的软元件重复。

软元件	功能	
	断开延时定时器单脉冲定时器	功能
D	断开延时定时器	占用
D+1	单脉冲定时器	闪烁（a触点）
D+2	占用	闪烁（b触点）
D+3	占用	闪烁（b触点）

（3）软元件的占用点数

D,D+1,D+3 在经过设定时间后变为 OFF。D+2 和定时器 D 被即时复位。

3.7.4 ALT 指令(交替输出)

输入为 ON 时，使位软元件反转（ON←→OFF）用的指令

1、指令格式

16bit 3 步		32bit		指令格式
ALT	ALTP			ALT D

操作数的数据类型如下表

操作数	内容	类型
D	交替输出的位软元件编号	位

操作数的对象软元件如下表

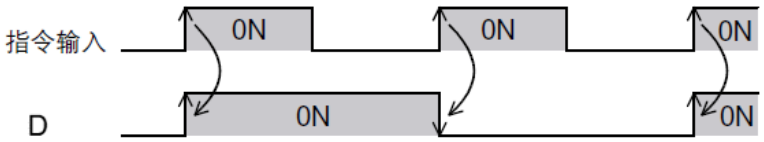
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D		●	●			●	●	●	●	▲				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D													●	

▲：Dx.y 不能变址修饰（V,Z）

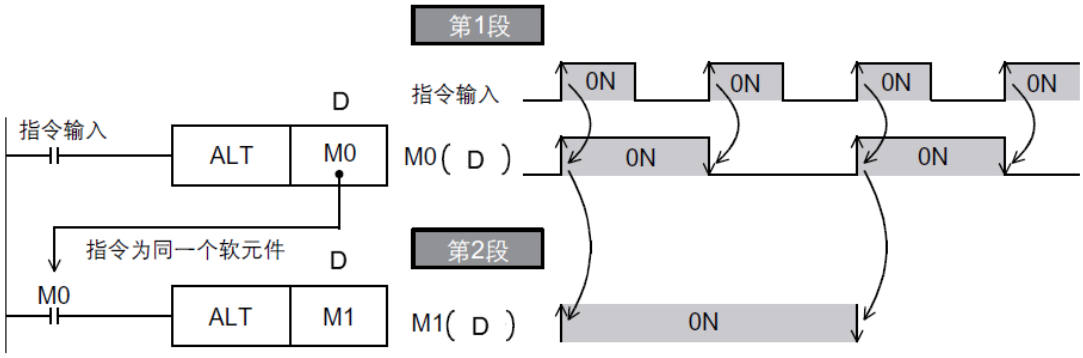
2、功能和动作说明

交替输出（1 段）

指令输入每次从 OFF 变为 ON 时，D 中指定的位软元件就执行 ON←→OFF 的反转。



分频输出（采用交替输出（2段））



3、程序举例

参考功能和动作说明

4、注意事项

（1）使用 ALT（连续执行型）时

使用 ALT 指令编程时，每个扫描周期都执行反转动作。

希望通过指令的 OFF/ON 使其反转动作时，请使用 ALTP 指令(脉冲执行型)，或是 LDP 指令触点(脉冲执行型)。

3.7.5 RAMP 指令(斜坡信号)

在开始（初始值）和结束（目标值）的 2 个值之间，按照指定的 n 次得到变化的数据的指令。

1、指令格式

16bit 9 步		32bit		指令格式
RAMP				STMR S1 S2 D n

操作数的数据类型如下表

操作数	内容	类型
S1	保存设定的斜坡初始值的软元件编号	BIN16 位
S2	保存设定的斜坡目标值的软元件编号	BIN16 位
D	保存斜坡的当前值数据的软元件编号	BIN16 位
n	斜坡的转移时间（扫描）	BIN16 位

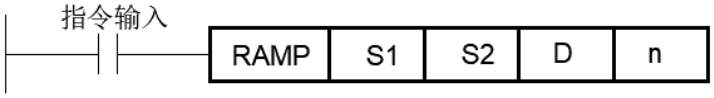
操作数的对象软元件如下表

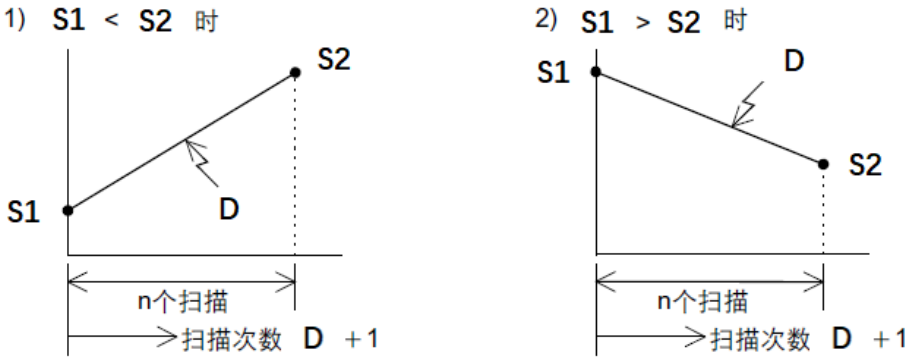
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
S2														
D														
m											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1							●	●	●	●			●	
S2							●	●	●	●			●	
D							●	●	●	●			●	
m							●	●	●	●				

2、功能和动作说明

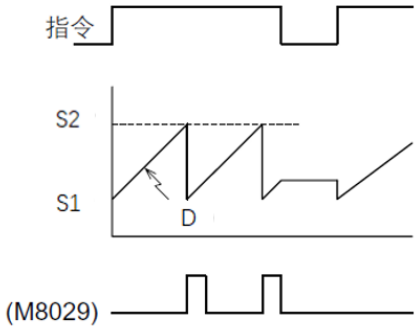
先指定开始的值 S1 和要结束的值 S2,当指令输入为 ON 后，在每个扫描周期中，将按照 n 中指定的次数进行等分的值依次加到 S1 中，然后将加得的值保存到 D 中。

将该指令和模拟量输出相结合，可以输出缓冲启动/停止指令。





- D+1 中保存扫描次数（0-n）。
- 才开始到结束所需时间，为运算周期*n 个扫描周期。
- 如果在动作过程中断开指令输入，则变为执行中断的状态（D 当前值数据保持；D+1 扫面次数被清除），再次置 ON 后，D 被清除，从 S1 开始重新动作。
- 转移结束后，指令执行结束标志位 M8029 动作，D 的值返回到 S1 的值。



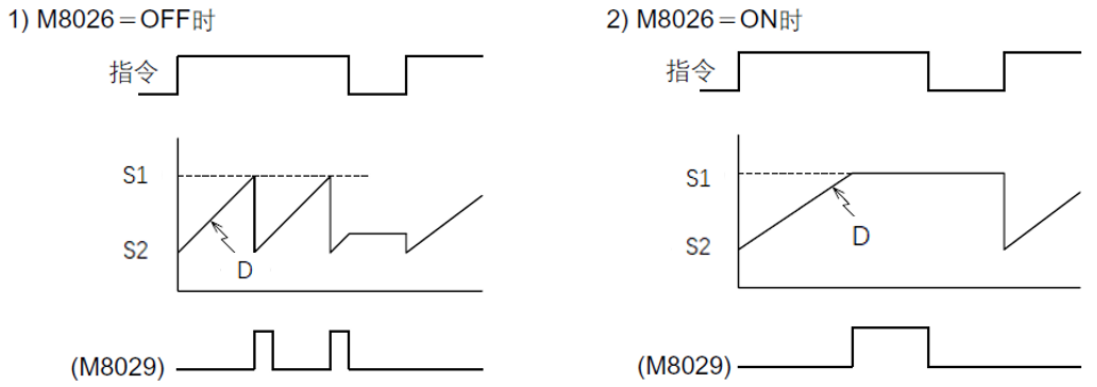
- 固定时间间隔下求出运算结果时（恒定扫描模式）

  将既定的扫描时间(比实际的扫描时间稍微长一点的值)写入 D8039，当 M8039 为 ON 时，可编程控制器就处于恒定扫描模式。

  例如，这个值指定了 20ms，n=100 次的时候，就表示在 20s 内 D 的值从 S1 变化到 S2.

模式标志位（M8026）的动作

根据模式标志位 M8026 的状态，D+1 的内容也做如下变更。

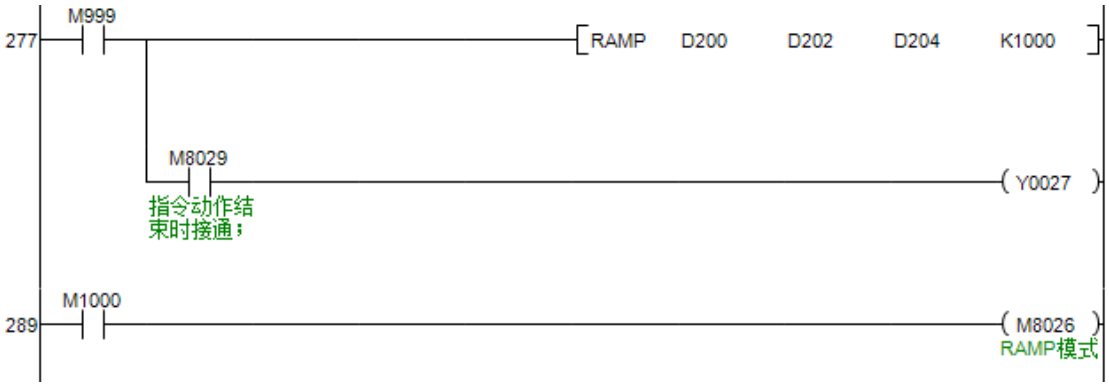


相关软元件

软元件	名称	内容
M8029	指令执行结束	n个扫描周期后，D ＝ S2 时置ON。
M8026※1	RAMP模式	请参考上面的模式标志位（M8026）的操作。

※1. RUN→ STOP时清除

3、程序举例



4、注意事项

D 中指定停电保持软元件时指令输入按原样置 ON，可编程控制器设置到 RUN 时，请先清除 D。

3.7.6 SORT 指令(数据排序)

该指令是用于将数据（行）和群数据（列）构成的数据表格，以指定的群数据（列）为标准，按照行单位将数据表格重新升序排列。在这个指令中，群数据（列）被保存在连续的软元件中。此外，数据（行方向）被保存在连续的软元件中。

1、指令格式

16bit 11 步	32bit	指令格式

SORT				STMR S m1 m2 D n

操作数的数据类型如下表

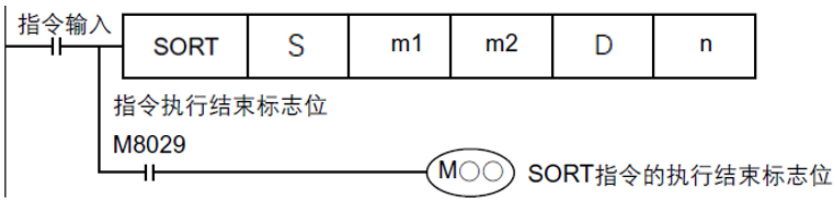
操作数	内容	类型
S	保存数据表格的软元件的起始编号[占用 m1*m2 点]	BIN16 位
m1	数据（行）数[1-32]	BIN16 位
m2	群数据（列）数[1-6]	BIN16 位
D	保存运算结果的软元件的起始编号[占用 m1*m2 点]	BIN16 位
n	作为排列标准的群数据（列）的列编号[1-m2]	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
m1											●	●		
m2											●	●		
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●				
m1														
m2														
D							●	●	●	●				
n							●	●	●	●				

2、功能和动作说明

在 S 开始的（m1*m2）的数据表格（排序前）中，以 n 列的群数据为标准，按照升序重新排列数据行，然后保存在从 D 开始的（m1*m2）点的数据表格（排序后）中。



下面以排序前 m1=3，m2=4 的情况为例来说明数据表格的结构。如图是排序后的数据表格，请将 S 改读成 D。

列号 / 行号		群数m2个（m2＝4时）			
		1	2	3	4
		管理编号	身高	体重	年龄
数据数 m1＝3 的情况	1	S	S +3	S +6	S +9
	2	S +1	S +4	S +7	S +10
	3	S +2	S +5	S +8	S +11

指令为 ON 的时候开始数据的排序，m1 个扫描后数据排序结束，指令执行结束标志位 M8029 为 ON。

相关软元件

软元件	名称	内容
M8029	指令执行结束	数据排序结束时为ON。

3、程序举例

以在“n=2（列号 2）”和“n=3（列号 3）”中执行下述的排序前的数据后，会如下所示动作。

动作例子是 16 位运算情况下的例子，32 位运算时，请使用 BIN32 位构成数据表格。

此外，如果现在第一列中输入管理编号等的连续编号，则可以根据其内容判断出原来所在的行号，因此非常方便。

排序前数据

行号 \ 列号		群数m2个（m2=4时）			
		1	2	3	4
		管理编号	身高	体重	年龄
数据数 m1=5 时	1	S	S +5	S +10	S +15
		1	150	45	20
	2	S +1	S +6	S +11	S +16
		2	180	50	40
	3	S +2	S +7	S +12	S +17
		3	160	70	30
	4	S +3	S +8	S +13	S +18
		4	100	20	8
	5	S +4	S +9	S +14	S +19
		5	150	50	45

1) 以 n=K2（列号2）执行指令时的排序结果

行号 \ 列号	1	2	3	4
	管理编号	身高	体重	年龄
1	D	D +5	D +10	D +15
	4	100	20	8
2	D +1	D +6	D +11	D +16
	1	150	45	20
3	D +2	D +7	D +12	D +17
	5	150	50	45
4	D +3	D +8	D +13	D +18
	3	160	70	30
5	D +4	D +9	D +14	D +19
	2	180	50	40

2) 以n＝K3（列号3）执行指令时的排序结果

列号 行号	1 管理编号	2 身高	3 体重	4 年龄
1	D	D +5	D +10	D +15
	4	100	20	8
2	D +1	D +6	D +11	D +16
	1	150	45	20
3	D +2	D +7	D +12	D +17
	2	180	50	40
4	D +3	D +8	D +13	D +18
	5	150	50	45
5	D +4	D +9	D +14	D +19
	3	160	70	30

4、注意事项

- 动作过程中，请勿改变操作数和数据内容。
- 再次执行时，请将指令输入 OFF 一次。
- 指令的使用次数的限制

  程序中仅可以使用一次。
- S 和 D 中指定了同一个软元件时

  原来的数据被改写成排序后的数据顺序。

  到执行结束为止，请特别注意不要改变 S 的内容。

3.8 外部设备 IO 指令

3.8.1 SEGD 指令(七段码译码)

数据译码后，点亮 7 段数码管（1 位数）的指令。

1、指令格式

16bit 5 步		32bit		指令格式
SEGD	SEGDP	\	\	SEGD S D

操作数的数据类型如下表

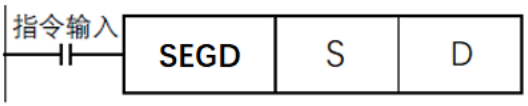
操作数	内容	类型
S	译码的起始字软元件	BIN16 位
D	保存 7 段码显示用数据的字软元件编号	BIN16 位

操作数的对象软元件如下表

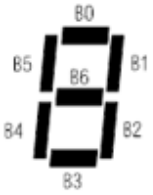
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

将 S 的低 4 位（1 位数）的 0-F（16 进制数）译码成 7 段码显示用的数据，并保存到 D 的低 8 位中。



7段码译码表

S					7段码的构成	D											显示数据
16进制数	b3	b2	b1	b0		B15	…	B8	B7	B6	B5	B4	B3	B2	B1	B0	
0	0	0	0	0		-		-	0	0	1	1	1	1	1	1	0
1	0	0	0	1		-		-	0	0	0	0	0	1	1	0	1
2	0	0	1	0		-		-	0	1	0	1	1	0	1	1	2
3	0	0	1	1		-		-	0	1	0	0	1	1	1	1	3
4	0	1	0	0		-		-	0	1	1	0	0	1	1	0	4
5	0	1	0	1		-		-	0	1	1	0	1	1	0	1	5
6	0	1	1	0		-		-	0	1	1	1	1	1	0	1	6
7	0	1	1	1		-		-	0	0	1	0	0	1	1	1	7
8	1	0	0	0		-		-	0	1	1	1	1	1	1	1	8
9	1	0	0	1		-		-	0	1	1	0	1	1	1	1	9
A	1	0	1	0		-		-	0	1	1	1	0	1	1	1	A
B	1	0	1	1		-		-	0	1	1	1	1	1	0	0	b
C	1	1	0	0		-		-	0	0	1	1	1	0	0	1	C
D	1	1	0	1		-		-	0	1	0	1	1	1	1	0	d
E	1	1	1	0		-		-	0	1	1	1	1	0	0	1	E
F	1	1	1	1		-		-	0	1	1	1	0	0	0	1	F

3、程序举例

无

4、注意事项

- 软元件的点数占用

  软元件 D 的输出开始的低 8 位被占用，高 8 位不变化。

3.8.2 ASC 指令(ASCII 数据输入)

将半角/英文数字字符串转换成 ASCII 码的指令

1、指令格式

16bit 11 步		32bit		指令格式
ASC	\	\	\	ASC S D

操作数的数据类型如下表

操作数	内容	类型
S	从计算机输入的 8 个字符的半角英文数字	字符串（仅 ASCII 码）
D	保存 ASCII 数据的起始字软元件编号	BIN16 位

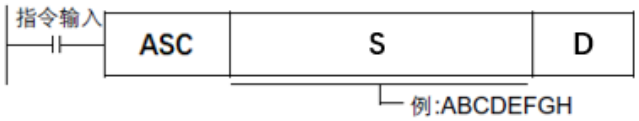
操作数的对象软元件如下表

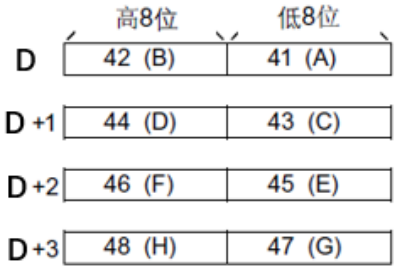
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														●
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S														
D					●	●	●	●	●	●			●	

2、功能和动作说明

将 S 中指定的半角、英文、数字字符串转换成 ASCII 码后，依次传送到 D 中。

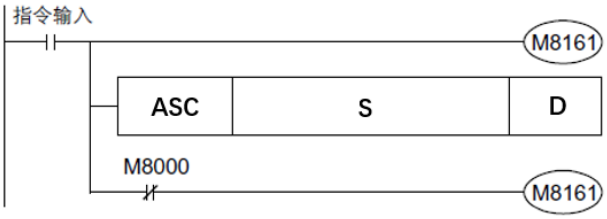
- 在 S 中处理 A-Z、0-9、符号的半角字符。（不处理全角字符串）
- 转换后的 ASCII 码按照低 8 位、高 8 位的顺序，每 2 个字符/1 字节地保存在 D 中。





扩展功能

M8161 置 ON 后，扩展功能变为有效，此时将 S 中指定的半角、英文、数字字符串转换成 ASCII 码，然后将其依次传送到 D 的低 8 位（1 个字节）中。



高8位为H00。

	D		S
	高8位	低8位	字符串
D	00	41	A
D +1	00	42	B
D +2	00	43	C
D +3	00	44	D
D +4	00	45	E
D +5	00	46	F
D +6	00	47	G
D +7	00	48	H

3、程序举例

无

4、注意事项

- 软元件的点数占用

  扩展功能 OFF 时

  D 中占用字符数除以 2 点。（不能整除时，则将小数点进位）

  扩展功能 ON 时

  D 中占用的点数和字符占用的数目相同。

扩展功能标志位 M8161 是与其他指令通用的标志位

在使用 ASC 指令时，请注意，在 ASC 指令的前面编写 M8161 ON 或 OFF 的程序，以不造成影响。

3.9 外部设备 SER

3.9.1 PRUN 指令(8 进制位传送)

将指定了位数的 S 和 D 的软元件编号作为 8 进制数处理，并传送数据。

1、指令格式

16bit 5 步		32bit		指令格式
PRUN	PRUNP	\	\	PRUN S D

操作数的数据类型如下表

操作数	内容	类型
S	位数指定（指定要素编号的最低位数请设置为 0）	BIN16 位/32 位
D	传送目标软元件编号（指定要素编号的最低位数请设置为 0）	BIN16 位/32 位

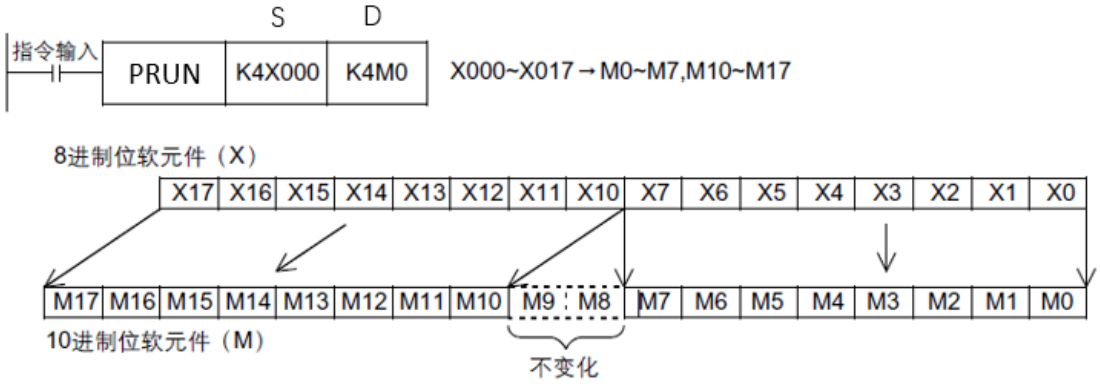
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●		●										●	
D		●	●										●	

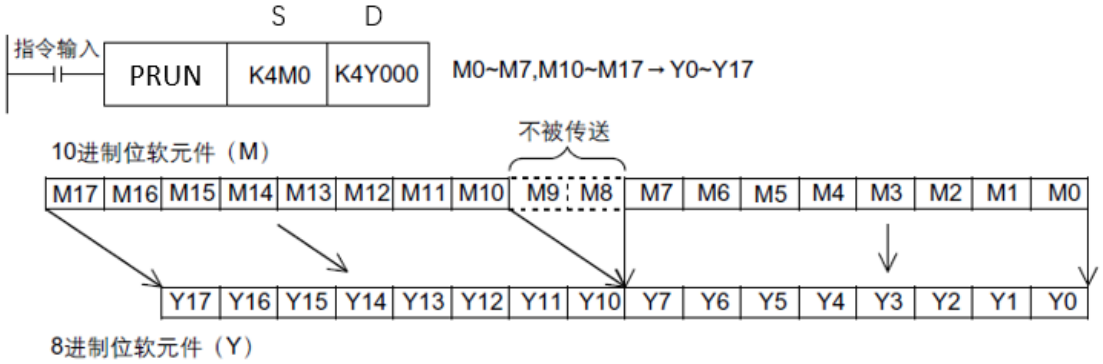
2、功能和动作说明

16 位运算

8进制位软元件→10进制位软元件

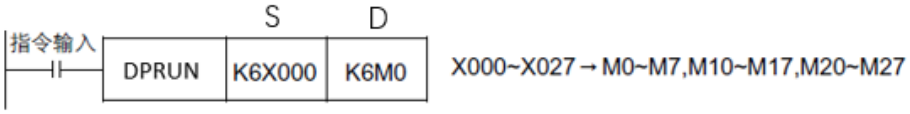


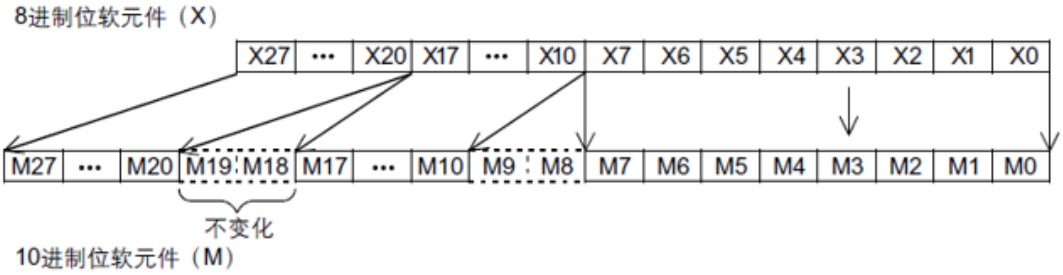
10进制位软元件→8进制位软元件



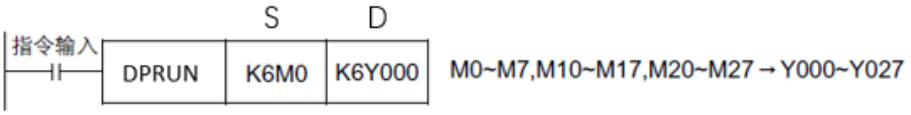
32 位运算

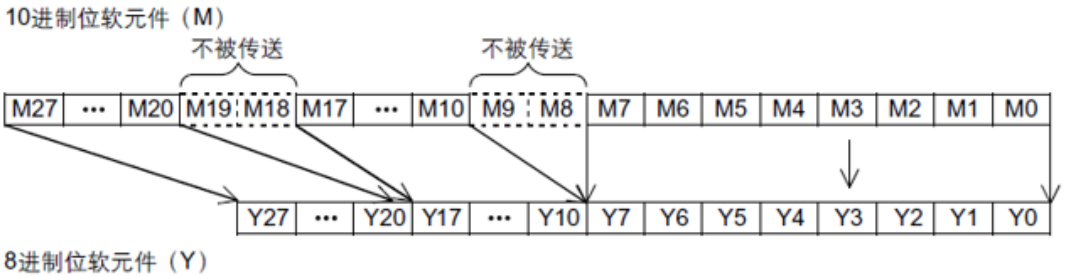
8进制位软元件→10进制位软元件





10进制位软元件→8进制位软元件





3、程序举例

无

4、注意事项

无

3.9.2 ASCI 指令(HEX 到 ASCII 码的转换)

将 HEX 转换成 ASCII 码的指令。

1、指令格式

	16bit 7 步		32bit		指令格式
ASCI	ASCI	ASCIP			PRUN S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存要转换的 HEX 的软元件的起始编号	BIN16 位
D	保存转换后的 ASCII 码的软元件的起始编号	字符串（仅 ASCII 码）
n	要转换 4 的 HEX 的字符数（位数）[设定范围：1-256]	BIN16 位

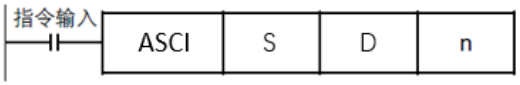
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●			●	
n							●	●	●	●				

2、功能和动作说明

16 位运算

将 S 开始的软元件中保存的 HEX 的 n 个字符（位数）转换成 ASCII 码，然后保存到 D 开始的软元件中。



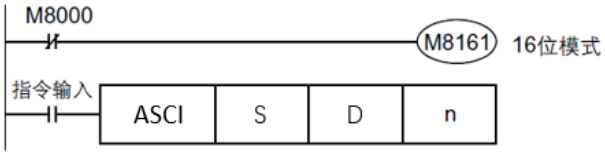
《16 位转换模式》M8161 为 OFF 时

将 S 开始的软元件中保存的各位 HEX 数据转换成 ASCII 码，然后传送到 D 开始的各软元件的高低 8 位（字节）中的指令。用 n 指定转换的位数（字符数）。

D 分低 8 位和高 8 位来保存 ASCII 数据。

此外 M8161 位多指令通用，使用 16 位转换模式时，请将 M8161 一直置位 OFF。

M8161 在 RUN 到 STOP 时被清除。



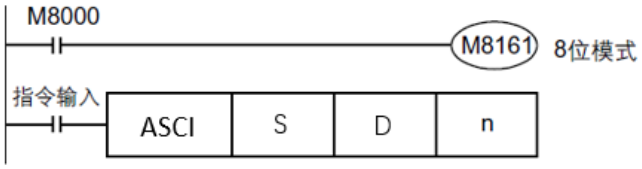
《8 位转换模式》M8161 为 ON 时

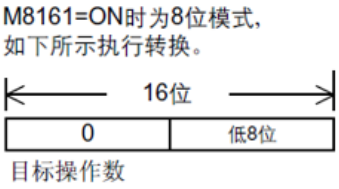
将 S 开始的软元件中保存的各位 HEX 数据转换成 ASCII 码，然后传送到 D 开始的各软元件的低 8 位（字节）中的指令。用 n 指定转换的位数（字符数）。

D 的高 8 位为 0。

此外 M8161 位多指令通用，使用 16 位转换模式时，请将 M8161 一直置位 ON。

M8161 在 RUN 到 STOP 时被清除。

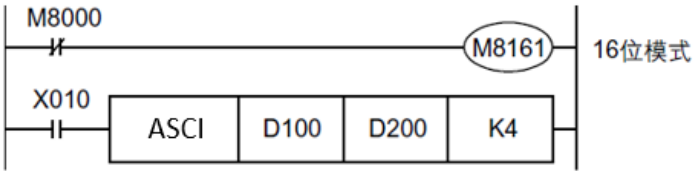




3、程序举例

- 《16 位转换模式》M8161 为 OFF 时

当为下述程序时，如下所示执行转换



S 开始的软元件

(D100)=0ABCH

(D101)=1234H

(D102)=5678H

指定位数（字符数）及转换结果

n / D	K1	K2	K3	K4	K5	K6	K7	K8	K9
D 200低	「C」	「B」	「A」	「0」	「4」	「3」	「2」	「1」	「8」
D 200高		「C」	「B」	「A」	「0」	「4」	「3」	「2」	「1」
D 201低			「C」	「B」	「A」	「0」	「4」	「3」	「2」
D 201高				「C」	「B」	「A」	「0」	「4」	「3」
D 202低					「C」	「B」	「A」	「0」	「4」
D 202高						「C」	「B」	「A」	「0」
D 203低			不变化。				「C」	「B」	「A」
D 203高								「C」	「B」
D 204低									「C」

n＝K 4时的位结构

D 100=0ABCH

0	0	0	0	1	0	1	0	1	0	1	1	1	1	0	0
0				A				B				C			

D 200

0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	0
「A」→41H								「0」→30H							

D 201

0	1	0	0	0	0	1	1	0	1	0	0	0	0	1	0
「C」→43H								「B」→42H							

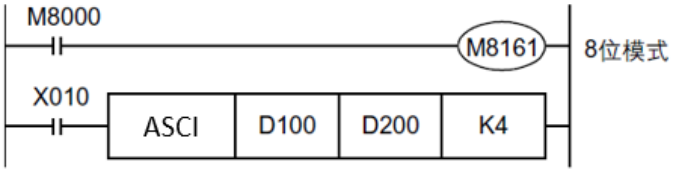
ASCII码

「0」=30H	「1」=31H	「5」=35H
「A」=41H	「2」=32H	「6」=36H
「B」=42H	「3」=33H	「7」=37H
「C」=43H	「4」=34H	「8」=38H

使用打印机等输出 BCD 数据时，需要在执行该指令前，先执行 BIN 到 BCD 的转换。

- 《8 位转换模式》M8161 为 ON 时

当为下述程序时，如下所示执行转换



S 开始的软元件

(D100)=0ABCH

(D101)=1234H

(D102)=5678H

指定位数（字符数）及转换结果

n / D	K1	K2	K3	K4	K5	K6	K7	K8	K9
D 200	「C」	「B」	「A」	「0」	「4」	「3」	「2」	「1」	「8」
D 201		「C」	「B」	「A」	「0」	「4」	「3」	「2」	「1」
D 202			「C」	「B」	「A」	「0」	「4」	「3」	「2」
D 203				「C」	「B」	「A」	「0」	「4」	「3」
D 204					「C」	「B」	「A」	「0」	「4」
D 205						「C」	「B」	「A」	「0」
D 206			不变化。				「C」	「B」	「A」
D 207								「C」	「B」
D 208									「C」

n＝K2时的位结构

D 100=0ABCH

0	0	0	0	1	0	1	0	1	0	1	1	1	1	0	0
0				A				B				C			

D 200=B的ACSII码=42 H

0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
								4				2			

D 201=C的ACSII码=43 H

0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
								4				3			

ASCII码

「0」=30H 「1」=31H 「5」=35H
「A」=41H 「2」=32H 「6」=36H
「B」=42H 「3」=33H 「7」=37H
「C」=43H 「4」=34H 「8」=38H

使用打印机等输出 BCD 数据时，需要在执行该指令前，先执行 BIN 到 BCD 的转换。

4、注意事项

无

3.9.3 HEX 指令(ASCII 到 HEX 的转换)

将 ASCII 转换成 HEX 码的指令。

1、指令格式

16bit 7 步		32bit		指令格式
HEX	HEXP	\	\	HEX S D n

操作数的数据类型如下表

操作数	内容	类型

S	保存要转换的 ASCII 码的软元件的起始编号	字符串（仅 ASCII 码）*1
D	保存转换后的 HEX 的软元件的起始编号	BIN16 位/32 位
n	要转换的 ASCII 码的字符数(位数)[设定范围：1-256]	BIN16 位

*1. 请仅将 ASCII 码设定为“0”－ “9”，“A”－“F”。

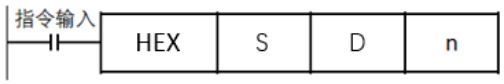
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●	●	●	●	
n							●	●	●	●				

2、功能和动作说明

16 位运算

将 S 开始的软元件中保存的 ASCII 码的 n 个字符（位数）转换成 HEX 代码，然后保存到 D 开始的软元件中。



《16 位转换模式》M8161 为 OFF 时

将保存在 S 的高低各 8 位（字节）中的 ASCII 字符码转换成 HEX 数据，然后按照每 4 位数的方式传送到 D 中。用 n 指定转换的位数（字符数）。

此外 M8161 位多指令通用，使用 16 位转换模式时，请将 M8161 一直置位 OFF。

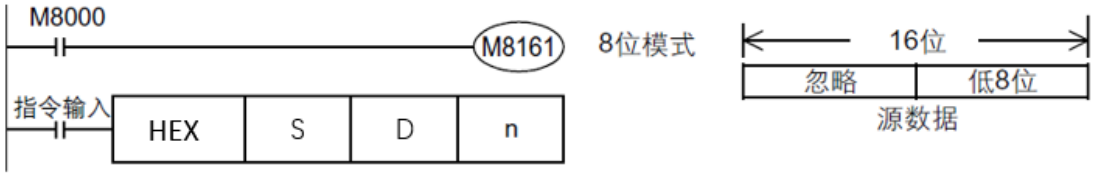
M8161 在 RUN 到 STOP 时被清除。

《8 位转换模式》M8161 为 ON 时

将保存在 S 的低 8 位（字节）中的 ASCII 字符码转换成 HEX 数据，然后按照每 4 位数的方式传送到 D 中。用 n 指定转换的位数（字符数）。

此外 M8161 位多指令通用，使用 16 位转换模式时，请将 M8161 一直置位 ON。

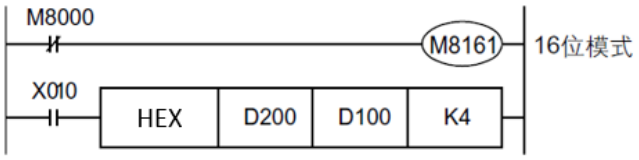
M8161 在 RUN 到 STOP 时被清除。



3、程序举例

- 《16 位转换模式》M8161 为 OFF 时

当为下述程序时，如下所示执行转换



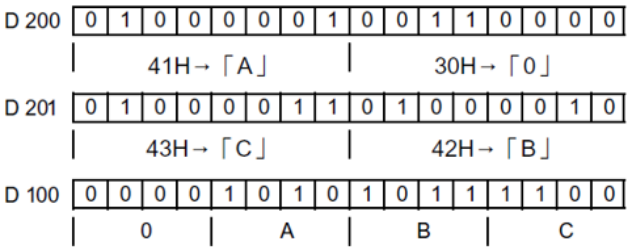
转换源数据

S	ASCII码	HEX 转换
D 200 低	30H	0
D 200 高	41H	A
D 201 低	42H	B
D 201 高	43H	C
D 202 低	31H	1
D 202 高	32H	2
D 203 低	33H	3
D 203 高	34H	4
D 204 低	35H	5

指定字符数及转换结果「・」为0。

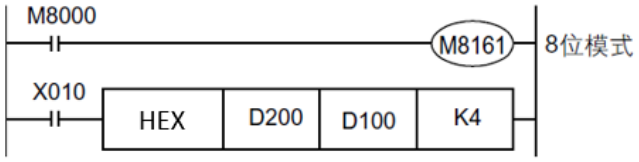
D / n	D 102	D 101	D 100
1	不变化		・・・0H
2			・・0AH
3			・0ABH
4			0ABCH
5		・・・0H	ABC1H
6		・・0AH	BC12H
7		・0ABH	C123H
8		0ABCH	1234H
9	・・・0H	ABC1H	2345H

n＝K4时



- 《8 位转换模式》M8161 为 ON 时

当为下述程序时，如下所示执行转换

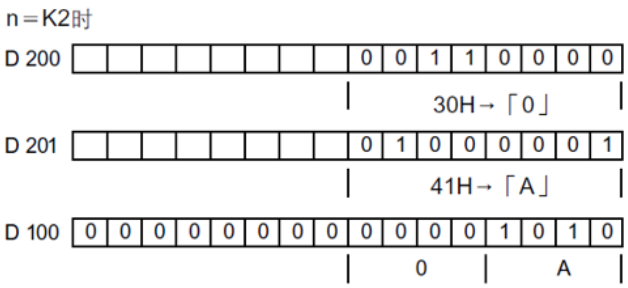


转换源数据

S	ASCII码	HEX 转换
D 200	30H	0
D 201	41H	A
D 202	42H	B
D 203	43H	C
D 204	31H	1
D 205	32H	2
D 206	33H	3
D 207	34H	4
D 208	35H	5

指定字符数及转换结果 「・」为0。

D n	D 102	D 101	D 100
1	不变化		···0H
2			··0AH
3			·0ABH
4			0ABCH
5		···0H	ABC1H
6		··0AH	BC12H
7		·0ABH	C123H
8		0ABCH	1234H
9	···0H	ABC1H	2345H



4、注意事项

输入数据为 BCD 时，需要在执行该指令前，先执行 BCD 到 BIN 的转换。

在 HEX 指令中，被保存在 S 中的数据如不是 ASCII 码，则会运算出错，不能进行 HEX 转换。尤其是当 M8161 为 OFF 时，在 S 的高 8 位中也需要事先保存 ASCII 码，请务必注意。

3.10 数据传送 2

3.10.1 ZPUSH 指令(变址寄存器的成批保存)

暂时保存变址寄存器 V0-V7,Z0-Z7 的当前值的指令。

要使暂时保存的当前值返回时，使用 ZPOP 指令。

1、指令格式

16bit 7 步		32bit		指令格式
ZPUSH	ZPUSHP			ZPUSH D

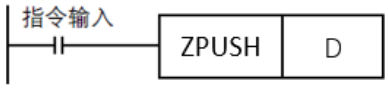
操作数的数据类型如下表

操作数	内容	类型
D	暂时保存变址寄存器 V0-V7,Z0-Z7 的当前值的软元件起始编号 D：成批保存次数 D+1 到 D+16 乘以成批保存次数：成批保存的数据保存的位置	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D							●	●	●	●				

2、功能和动作说明

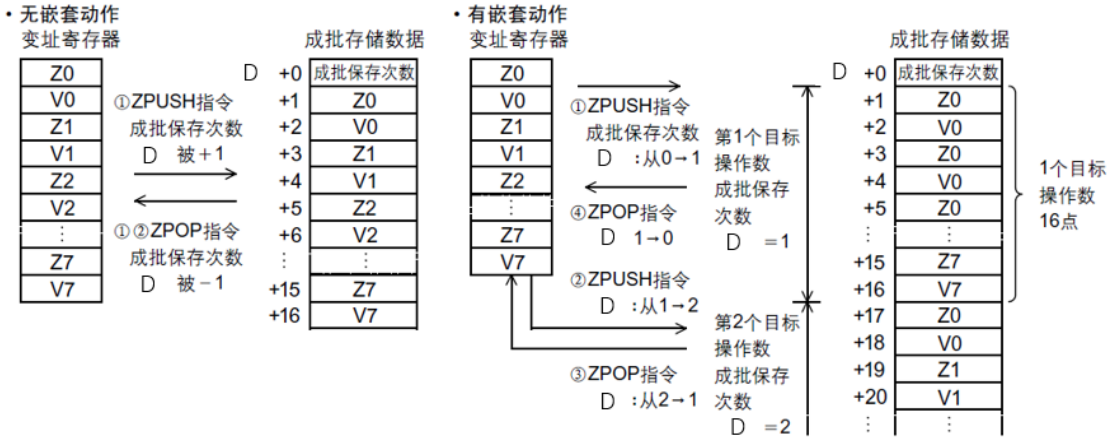


- 将变址寄存器 V0-V7, Z0-Z7 的内容成批保存到 D 开始的软元件中。成批保存了变址寄存器的内容后，成批保存次数 D 就被+1.
- 使用 ZPOP 指令使数据返回。

  成对使用 ZPUSH，ZPOP 指令。
- 通过 D 指定相同的软元件，可以嵌套使用 ZPUSH, ZPOP 指令。

  此时，每次执行 ZPUSH 指令时，D 开始使用的区域会每次增加 16 点。

  因此，请事先确保嵌套中使用的次数的区域。
- D 以后被成批保存的数据的结构如下图所示。

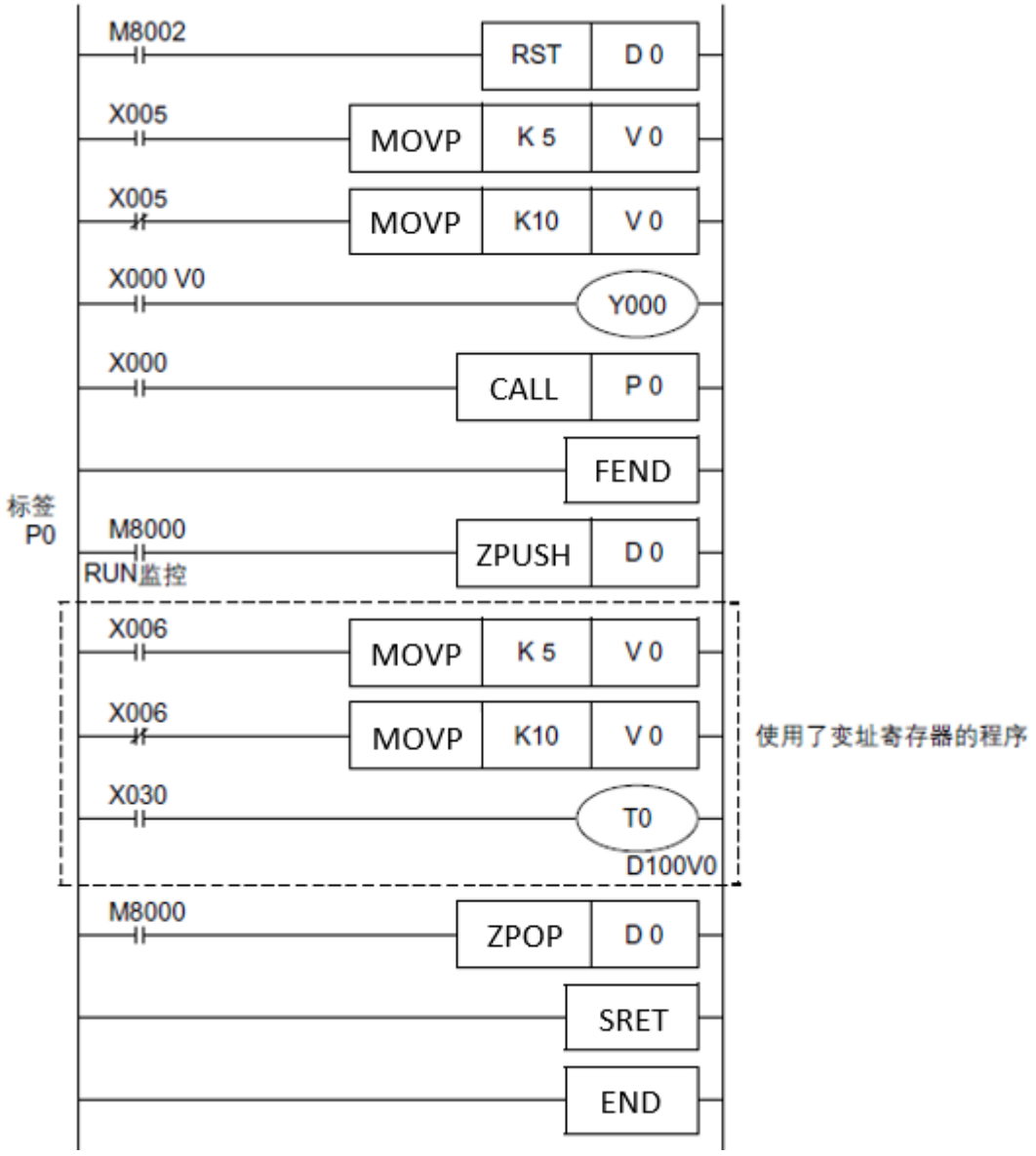


相关指令

指令	内容
ZPOP	使因 ZPUSH 指令而暂时成批保存的变址寄存器V0～V7、Z0～Z7恢复的指令。

3、程序举例

在指针 P0 以后的子程序中使用了变址寄存器，在执行子程序之前，先将变址寄存器 V0-V7,Z0-Z7 的内容成批保存到 D0 以后的程序。



4、注意事项

- 无嵌套动作时，执行 ZPUSH 指令前请清除成批保存次数 D
- 有嵌套动作时，在第一次执行前清除成批保存次数 D

3.10.2 ZPOP 指令(变址寄存器的恢复)

将使用 ZPUSH 指令进行暂时成批保存的变址寄存器 V0-V7,Z0-Z7 中的内容恢复回去

1、指令格式

16bit 7 步		32bit		指令格式
ZPOP	ZPOPP			ZPOP D

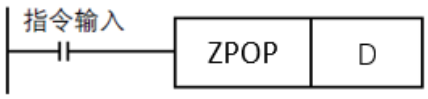
操作数的数据类型如下表

操作数	内容	类型
D	暂时保存变址寄存器 V0-V7,Z0-Z7 内容的软元件起始编号 D：成批保存次数 D+1 到 D+16 乘以成批保存次数：成批存储的数据的保存位置	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D							●	●	●	●				

2、功能和动作说明



- 将使用 ZPUSH 指令被暂时成批保存到 S 开始的软元件中的变址寄存器 V0-V7,Z0-Z7 的内容恢复到原来的变址寄存器中，变址寄存器的内容恢复后，成批保存次数 D 被-1.
- 使用 ZPUSH 指令，暂时成批保存数据。

成对使用 ZPUSH，ZPOP 指令。

相关指令

指令	内容
ZPUSH	暂时成批保存变址寄存器V0～V7、Z0～Z7的当前值的指令。

3、程序举例

无

4、注意事项

无

3.11 浮点运算指令

3.11.1 ECMP 指令(2 进制浮点数比较)

比较 2 个数据(2 进制浮点数)，将结果(大于、等于或小于)输出到位软元件(3 点)中的指令。

1、指令格式

16bit		32bit 13 步		指令格式
\	\	DECMP	DECMPP	DECMP S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	保存要比较的 2 进制浮点数数据的软元件编号	实数(2 进制)
S2	保存要比较的 2 进制浮点数数据的软元件编号	实数(2 进制)
D	输出结果的起始位软元件编号(占用 3 点)	位

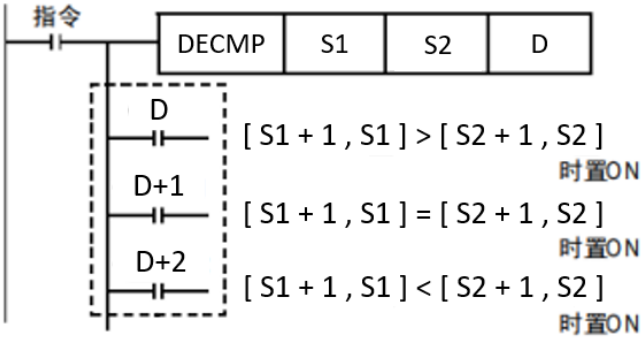
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●	●	
S2											●	●	●	
D		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针

	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1							●	●	●	●			●	
S2							●	●	●	●			●	
D													●	

2、功能和动作说明

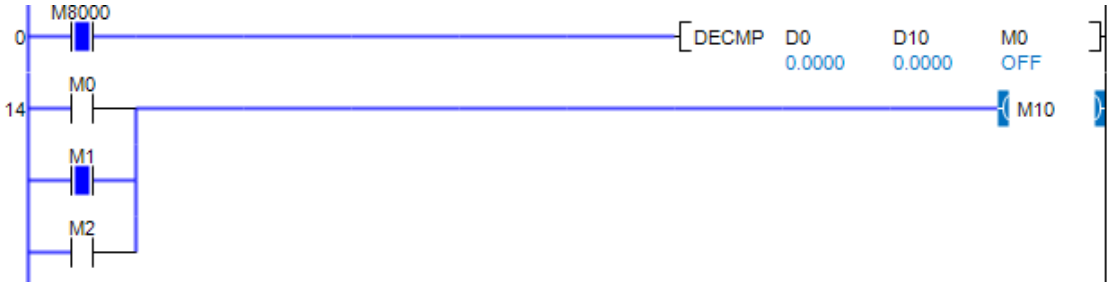
32 位运算指令，将比较值（S1+1，S1）和比较源（S2+1，S2）作为浮点数数据进行比较，然后根据比较的结果(小于、等于、大于)将 D、D+1、D+2 中的任意一位置 ON。（S1 +1，S1）、（S2 +1，S2）中指定了常数(K、H)时，会自动将数值从 BIN 转换成 2 进制浮点数后再处理。



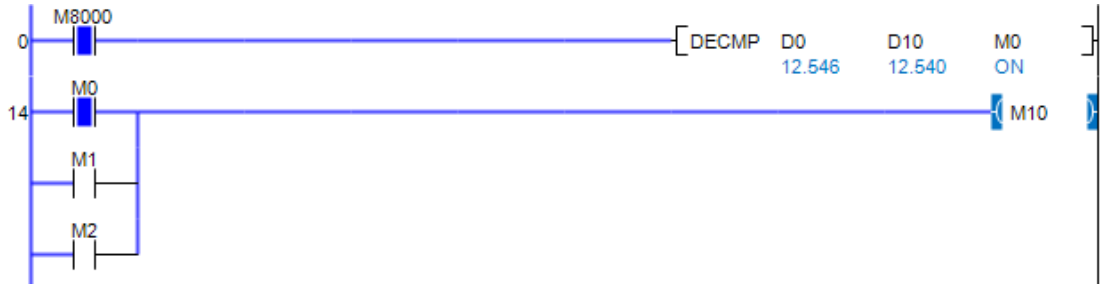
3、程序举例

当 PLC 为 RUN 的时候，比较 D0 与 D10 的值。

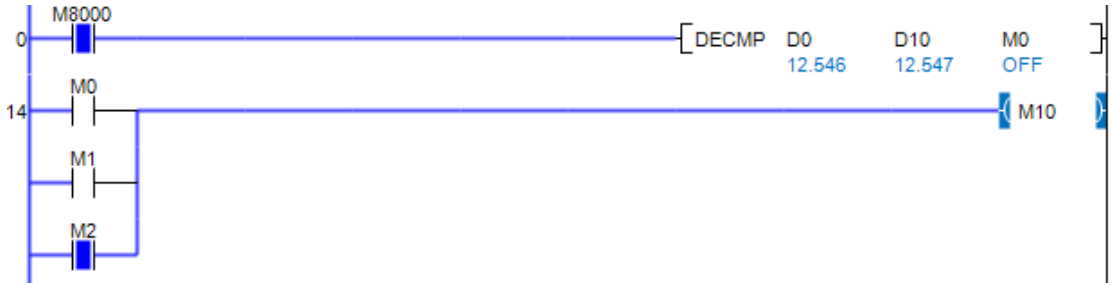
D0,D1 = D10,D11



D0,D1 > D10,D11



D0,D1<D10,D11



4、注意事项

（1）D 占用 3 点，（D，D+1，D+2），注意不要与其他用途的软元件重复。

（2）即使指令输入 OFF，不执行 DECMP 指令，D～D+2 也能保持指令输入 OFF 之前的状态。

3.11.2 EZCP 指令(2 进制浮点数区间比较)

将上下 2 点的比较范围和数据(2 进制浮点数)进行比较，根据其结果输出到位软元件(3 点)中的指令。

1、指令格式

16bit		32bit 17 步		指令格式
\	\	DEZCP	DEZCPP	DEZCP S1 S2 S D

操作数的数据类型如下表

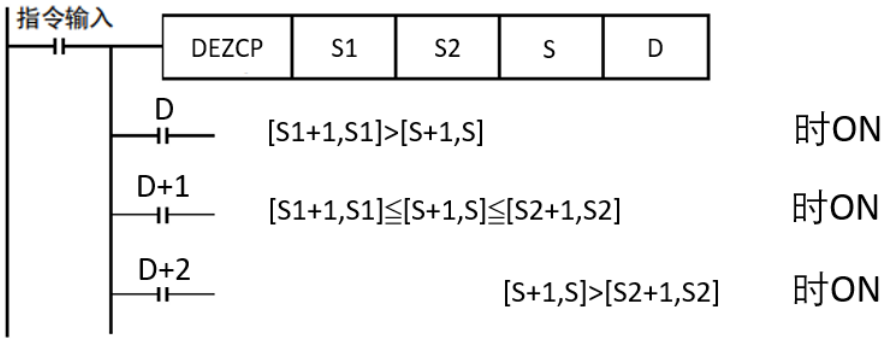
操作数	内容	类型
S1	保存要比较的 2 进制浮点数数据的软元件编号	实数(2 进制)
S2	保存要比较的 2 进制浮点数数据的软元件编号	实数(2 进制)
S	保存要比较的 2 进制浮点数数据的软元件编号	实数(2 进制)
D	输出结果的起始位软元件编号(占用 3 点)	位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●	●	
S2											●	●	●	
S											●	●	●	
D		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1							●	●	●	●			●	
S2							●	●	●	●			●	
S							●	●	●	●			●	
D							●	●	●	●			●	

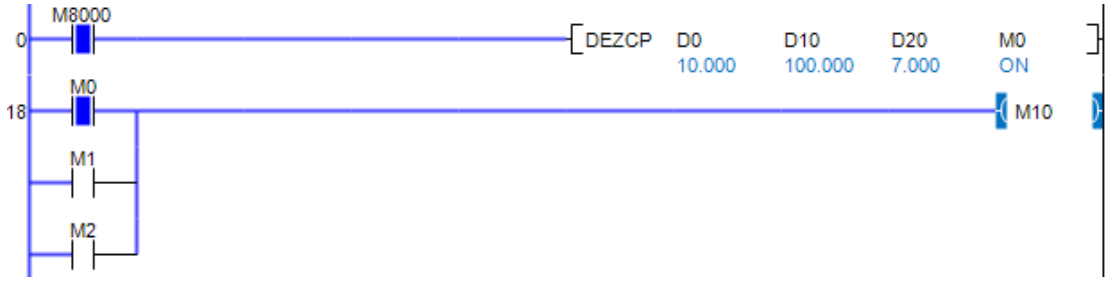
2、功能和动作说明

32 位运算指令，将比较值(S1+1，S1)、(S2+1，S2)和比较源(S+1，S)作为浮点数数据进行比较，然后根据比较的结果(小于、等于、大于)将 D、D+1、D+2 中的任意一位置 ON。在(S1+1，S1)、(S2+1，S2)、(S+1，S)中指定了常数(K、H)时，会自动将数值从 BIN 转换成 2 进制浮点数后再处理。

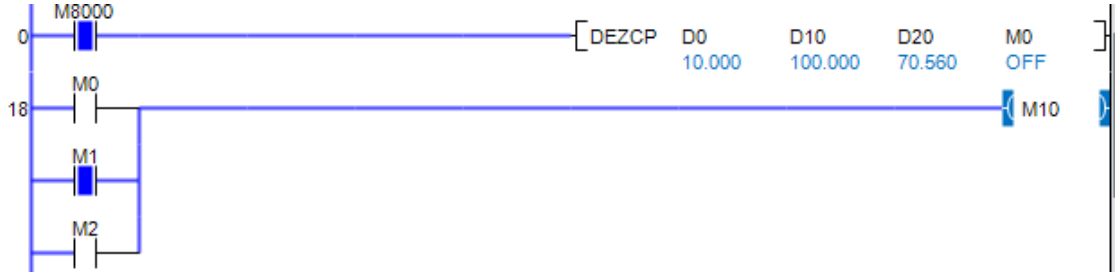


3、程序举例

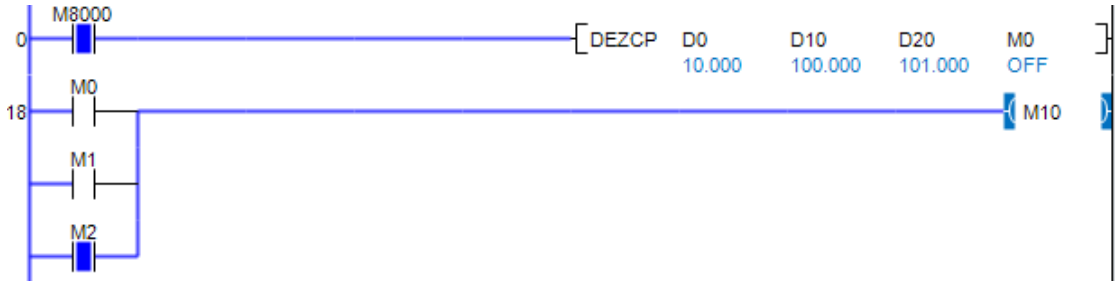
[S1+1,S1]>[S+1,S]



[S1+1,S1]≦[S+1,S]≦[S2+1,S2]



[S+1,S]>[S2+1,S2]



4、注意事项

（1）比较数据的大小关系请设置为(S1+1,S)≦(S2+1，S2)。(S1+1,S) ＞(S2+1，S2) 的情况时，(S2+1，S2) 的数值视为与(S1+1,S)相同，从而进行比较。

（2）即使指令输入 OFF，不执行 DECMP 指令，D～D+2 也能保持指令输入 OFF 之前的状态。

3.11.3 EMOV 指令(2 进制浮点数的传送)

传送 2 进制浮点数数据的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DEMOV	DEMOVP	DEMOV S D

操作数的数据类型如下表

操作数	内容	类型
S	传送源的 2 进制浮点数数据，或是保存数据的软元件编号	实数(2 进制)
D	保存 2 进制浮点数数据的软元件编号	实数(2 进制)

操作数的对象软元件如下表

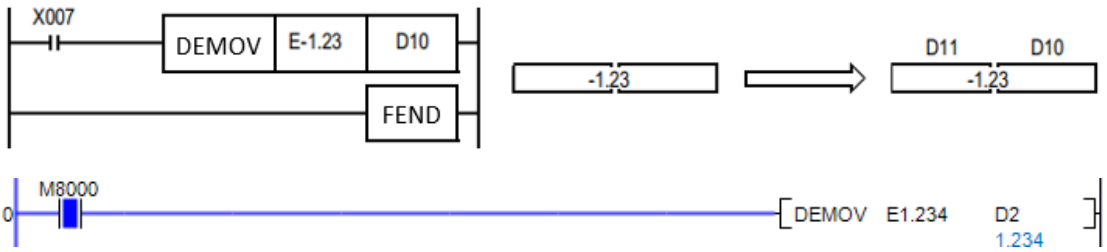
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

32 位运算指令，将传送源（S+1，S）的内容(2 进制浮点数数据)传送到（D+1,D）中。此外还可以在 S 中直接指定实数(E)。

3、程序举例

X007 为 ON 时，将实数-1.23 保存到 D11、D10 中的程序。



4、注意事项

在浮点数运算中，都以2进制浮点数执行。
但是，由于2进制浮点数本身是不易理解的数值(专用的监控方法)，所以转换成10进制浮点数运算后，可以便于在外围设备上进行监控等。

3.11.4 EBCD 指令(2 进制浮点数到 10 进制浮点数的转换)

将软元件中的 2 进制浮点数转换成→10 进制浮点数的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DEBCD	DEBCDP	DEBCD S D

操作数的数据类型如下表

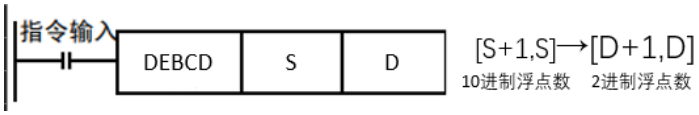
操作数	内容	类型
S	保存 2 进制浮点数数据的数据寄存器编号	实数(2 进制)
D	保存被转换的 10 进制浮点数数据的数据寄存器编号	实数(10 进制)

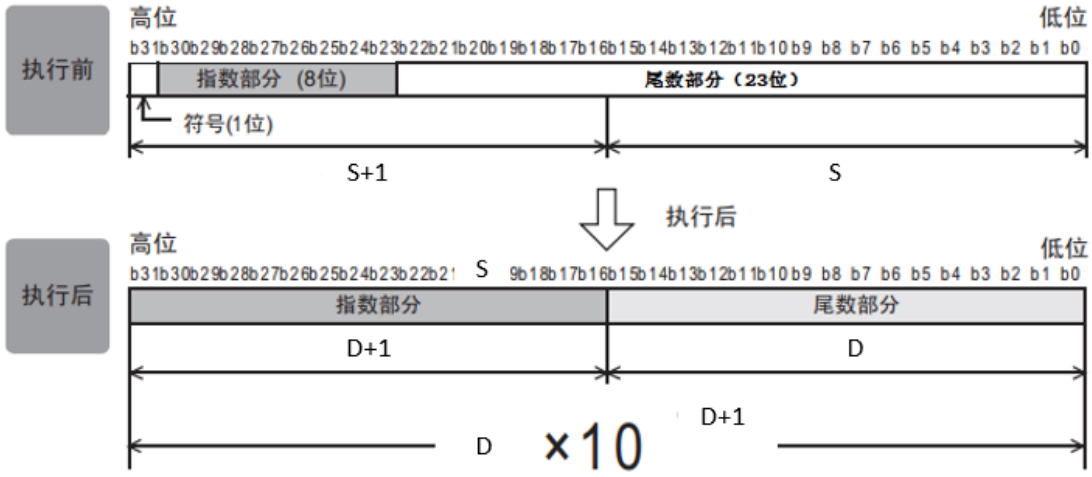
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

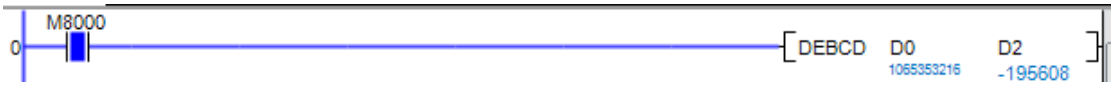
32 位运算指令，（S+1，S）的 2 进制浮点数转换成 10 进制浮点数后，保存到（D+1，D）中。





3、程序举例

当 PLC 为 RUN 时，将 D0 的值转成十进制浮点数。



4、注意事项

在浮点数运算中，都以 2 进制浮点数执行。但是，由于 2 进制浮点数本身是不易理解的数值(专用的监控方法)，所以转换成 10 进制浮点数运算后，可以便于在外围设备上进行监控等。

3.11.5 EBIN 指令(10 进制浮点数到 2 进制浮点数的转换)

将软元件中的 10 进制浮点数转换成 2 进制浮点数的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DEBIN	DEBINP	DEBIN S D

操作数的数据类型如下表

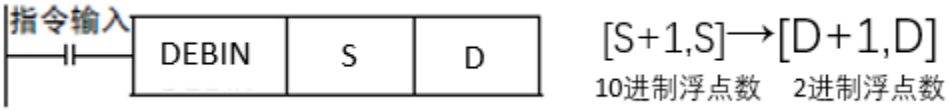
操作数	内容	类型
S	保存 10 进制浮点数数据的数据寄存器编号	实数(10 进制)
D	保存被转换的 2 进制浮点数数据的数据寄存器编号	实数(2 进制)

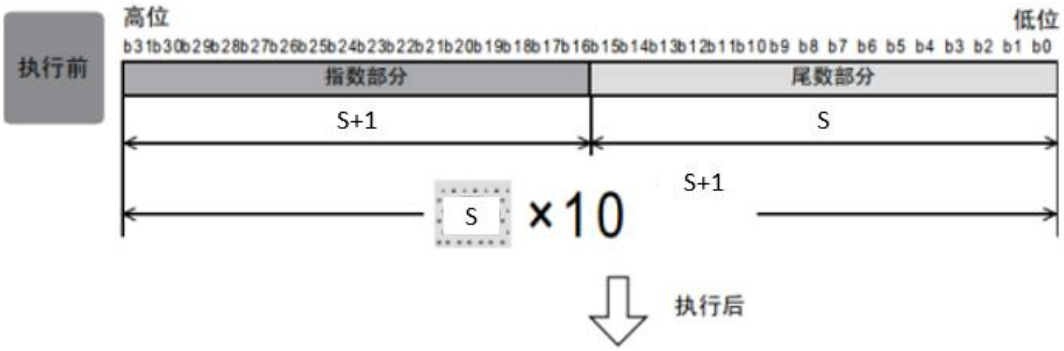
操作数的对象软元件如下表

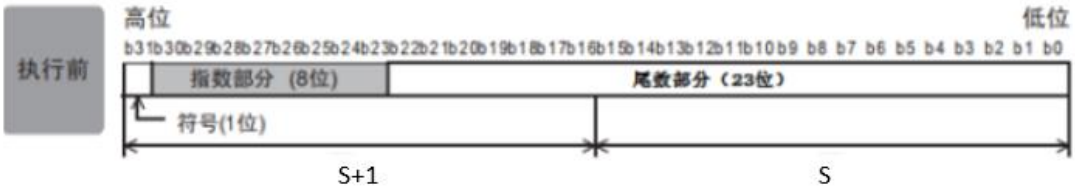
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

32 位运算指令，（S+1，S）的 10 进制浮点数转换成 2 进制浮点数后，保存到（D+1，D）中。





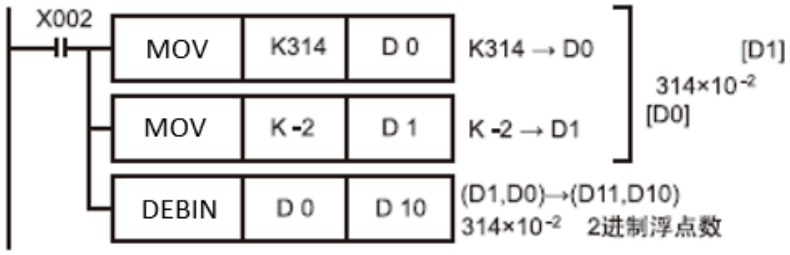


3、程序举例

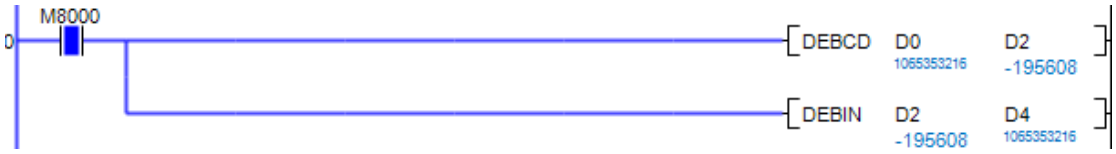
使用DEBIN指令后，包含了小数点的数值等可以被直接转换为2进制浮点数。

例：3.14的2进制浮点数转换

3.14=314×10-2(10 进制浮点数)



当 PLC 为 RUN 的时候，将 D2 的值转成二进制浮点数。



4、注意事项

注意 2 进制浮点数的存储方式与 10 进制的区别。

3.11.6 EADD 指令(2 进制浮点数加法运算)

2 个 2 进制浮点数加法运算的指令。

1、指令格式

16bit		32bit 13 步		指令格式
\	\	DEADD	DEADDP	DEADD S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	保存进行加法运算的 2 进制浮点数数据的字软元件编号	实数(2 进制)
S2	保存进行加法运算的 2 进制浮点数数据的字软元件编号	实数(2 进制)
D	保存加法运算后的 2 进制浮点数数据	实数(2 进制)

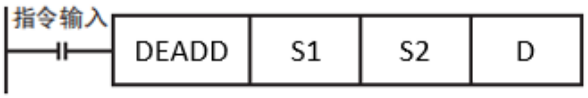
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”

S1											●	●	●	
S2											●	●	●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1							●	●	●	●			●	
S2							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

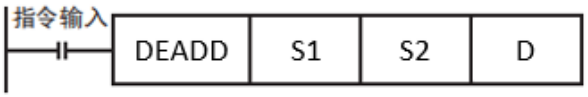
32 位运算指令，将（S1+1，S1）和（S2+1，S2)的 2 进制浮点数数据进行加法运算，并将其运算结果以 2 进制浮点数形式传送到(D+1，D)中。在(S1+1，S1)和(S2+1，S2)中指定了常数(K、H)时，数值会自动被转换成 2 进制浮点数。



[S1+1,S1] +[S2+1,S2]→[D+1,D]
二进制浮点数 二进制浮点数 二进制浮点数

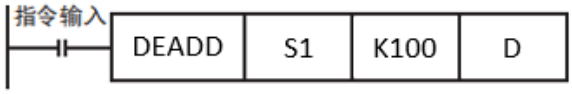
3、程序举例

将[S1+1，S1]和[S2+1，S2]的2进制浮点数数据进行加法运算，并将其运算结果以2进制浮点数形式传送到[D+1，D]中。



[S1+1,S1] +[S2+1,S2]→[D+1,D]
二进制浮点数 二进制浮点数 二进制浮点数

在[S1+1，S1]和[S2+1，S2]中指定了常数(K、H)时，数值会自动被转换成 2 进制浮点数。



[S1+1,S1] +[K100]→[D+1,D]
二进制浮点数 自动转化为二进制浮点数 二进制浮点数

4、注意事项

[S1+1，S1]以及[S2+1，S2]和[D+1，D]中也可以指定同一个软元件编号。
此时，如使用连续执行型的指令(DEADD)，则每一个运算周期其加法运算的结果都会改变，请务必注意。

3.11.7 ESUB 指令(2 进制浮点数减法运算)

2 个 2 进制浮点数减法运算的指令。

1、指令格式

16bit		32bit 13 步		指令格式
\	\	DESUB	DESUBP	DESUB S1 S2 D

操作数的数据类型如下表

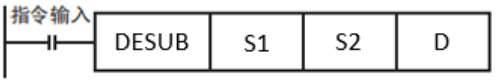
操作数	内容	类型
S1	保存进行减法运算的 2 进制浮点数数据的字软元件编号	实数(2 进制)
S2	保存进行减法的 2 进制浮点数数据的字软元件编号	实数(2 进制)
D	保存减法运算后的 2 进制浮点数数据	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●	●	
S2											●	●	●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1							●	●	●	●			●	
S2							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

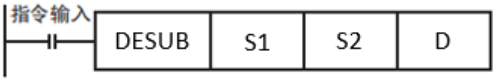
32 位运算指令，将[S1+1，S1]减去[S2+1，S2]的 2 进制浮点数数据，并将其运算结果以 2 进制浮点数形式传送到[D+1，D]中。在[S1+1，S1]和[S2+1，S2]中指定了常数(K、H)时，数值会自动被转换成 2 进制浮点数。



[S1+1,S1] -[S2+1,S2]→[D+1,D]
二进制浮点数 二进制浮点数 二进制浮点数

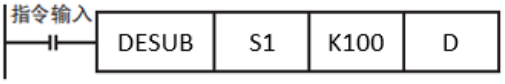
3、程序举例

从[S1+1， S1]中减去[S2+1，S2]的2进制浮点数数据，并将其运算结果以2进制浮点数形式传送到[D+1，D]中。



[S1+1,S1] -[S2+1,S2]→[D+1,D]
二进制浮点数 二进制浮点数 二进制浮点数

在[S1+1，S1]和[S2+1，S2]中指定了常数(K、H)时，数值会自动被转换成 2 进制浮点数。



[S1+1,S1] -[K100]→[D+1,D]
二进制浮点数 自动转化为 二进制浮点数
二进制浮点数

4、注意事项

[S1+1，S1]以及[S2+1，S2]和[D+1，D]中也可以指定同一个软元件编号。

此时，如使用连续执行型的指令(DESUB)，则每一个运算周期其减法运算的结果都会改变，请务必注意。

3.11.8 EMUL 指令(2 进制浮点数乘法运算)

2 个 2 进制浮点数乘法运算的指令。

1、指令格式

16bit		32bit 13 步		指令格式
\	\	DEMUL	DEMULP	DEMUL S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	保存进行乘法运算的 2 进制浮点数数据的字软元件编号	实数(2 进制)
S2	保存进行乘法的 2 进制浮点数数据的字软元件编号	实数(2 进制)
D	保存乘法运算后的 2 进制浮点数数据	实数(2 进制)

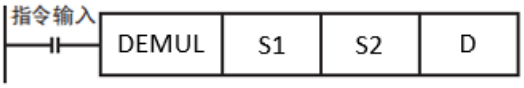
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●	●	
S2											●	●	●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P

S1							●	●	●	●			●	
S2							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

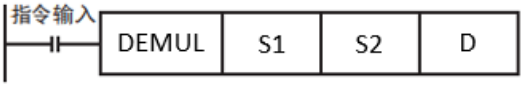
32 位运算指令，将[S1+1，S1]和[S2+1，S2]的 2 进制浮点数数据相乘，并将其运算结果以 2 进制浮点数形式传送到[D+1，D]中。在[S1+1，S1]和[S2+1，S2]中指定了常数(K、H)时，数值会自动被转换成 2 进制浮点数。



[S1+1,S1] x[S2+1,S2]→[D+1,D]
二进制浮点数 二进制浮点数 二进制浮点数

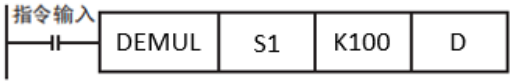
3、程序举例

将[S1+1，S1]和[S2+1，S2]的2进制浮点数数据相乘，并将其运算结果以2进制浮点数形式传送到[D+1，D]中。



[S1+1,S1] x[S2+1,S2]→[D+1,D]
二进制浮点数 二进制浮点数 二进制浮点数

在[S1+1，S1]和[S2+1，S2]中指定了常数(K、H)时，数值会自动被转换成2进制浮点数。



[S1+1,S1] X[K100]→[D+1,D]
二进制浮点数 自动转化为 二进制浮点数
二进制浮点数

4、注意事项

运算结果超过单精度浮点数范围时，对应的标志位置位

软元件	名称	内容	
		条件	动作
M8020	零位	转换结果真的为零(尾数部分为“0”时)	零位标志位(M8020)为 ON。
M8021	借位	转换结果的绝对值＜2^-126	D 的值小于 32 位实数的最小值 2^-126)部分被舍去，借位标志位(M8021)为 ON。
M8022	进位	转换结果的绝对值≧2^128	的值大于 32 位实数的最大值(2^128)部分被舍去，进位标志位(M8022)为 ON。

3.11.9 EDIV 指令(2 进制浮点数除法运算)

2 个 2 进制浮点数除法运算的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DEDIV	DEDIVP	DEDIV S1 S2 D

操作数的数据类型如下表

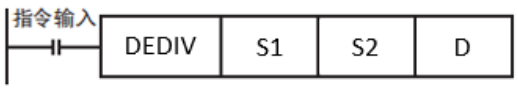
操作数	内容	类型
S1	保存进行除法运算的 2 进制浮点数数据的字软元件编号	实数(2 进制)
S2	保存进行除法的 2 进制浮点数数据的字软元件编号	实数(2 进制)
D	保存除法运算后的 2 进制浮点数数据	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●	●	
S2											●	●	●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1							●	●	●	●			●	
S2							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

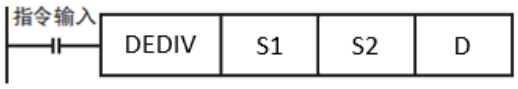
32 位运算指令，将[S1+1，S1]和[S2+1，S2]的 2 进制浮点数数据相除，并将其运算结果以 2 进制浮点数形式传送到[D+1，D]中。在[S1+1，S1]和[S2+1，S2]中指定了常数(K、H)时，数值会自动被转换成 2 进制浮点数。



被除数 除数
[S1+1,S1] ÷[S2+1,S2]→[D+1,D]
二进制浮点数 二进制浮点数 二进制浮点数

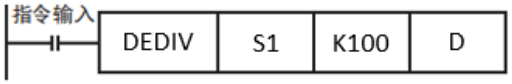
3、程序举例

将[S1+1，S1]和[S2+1，S2]的2进制浮点数数据相除，并将其运算结果以2进制浮点数形式传送到[D +1，D]中。



被除数　　除数
[S1+1,S1] ÷[S2+1,S2]→[D+1,D]
二进制浮点数 二进制浮点数 二进制浮点数

在[S1+1，S1]和[S2 +1，S2]中指定了常数(K、H)时，数值会自动被转换成2进制浮点数。



被除数　　除数
[S1+1,S1] ÷[　K100　]→[D+1,D]
二进制浮点数　自动转化为　二进制浮点数
二进制浮点数

4、注意事项

运算结果超过单精度浮点数范围时，对应的标志位 M8020，M8021，M8022 的动作。

3.11.10 EXP 指令(2 进制浮点数指数运算)

该指令是以 e(2.71828)为底的指数运算指令。

1、指令格式

16bit		32bit　9 步		指令格式
\	\	DEXP	DEXPP	DEXP S D

操作数的数据类型如下表

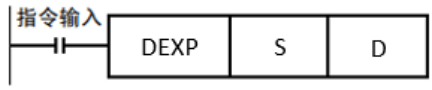
操作数	内容	类型
S	保存执行指数运算的 2 进制浮点数数据的软元件起始编号	实数(2 进制)
D	保存运算结果的软元件起始编号	实数(2 进制)

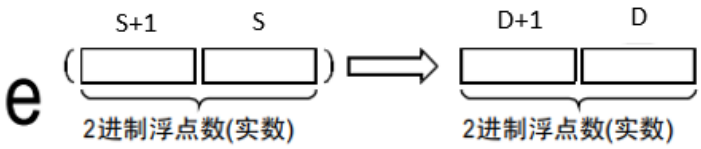
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

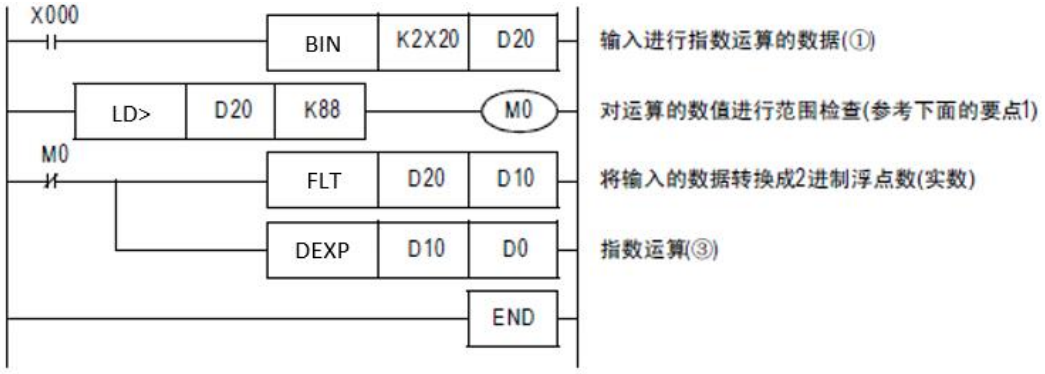
32 位运算指令，以[S+1，S1]为指数做运算，并将其运算结果保存到[D+1，D]中。此外，可以在 S 中直接指定实数。

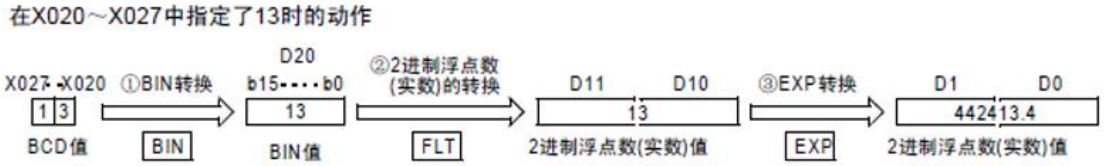




3、程序举例

X000为ON后，对X020～X027中以BCD2位数形式设定的数值进行指数运算，并且保存在2进制浮点数D0、D1中的程序。





1) 由于$loge2^{128} = 88.71$，因此当X020～X027的BCD值为88以下时，此时运算结果不满2^{128}。设定了89以上的数值时，会发生运算错误，因此设定了89以上的数值时，M0置ON，从而不执行运算。

2) 从自然对数向常用对数的转换在CPU中，执行自然对数的运算。

要求出常用对数值时，请在[S+1,S]中指定用0.4342945分割常用对数的值。

$$10^{X}=e^{\frac{X}{0.4342945}}$$

当PLC为RUN时，以[S+1，S1]为指数做运算，并将其运算结果保存到[D+1，D]中。

4、注意事项

结果不在 2^{-126}≦｜运算结果｜＜2^{128} 范围内，报运算错误 6706。

3.11.11 LOGE 指令(2 进制浮点数自然对数运算)

该指令执行自然对数运算。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DLOGE	DLOGEP	DLOGE S D

操作数的数据类型如下表

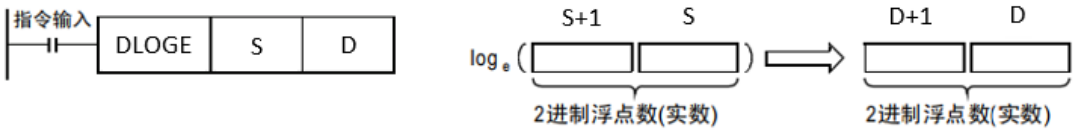
操作数	内容	类型
S	保存进行自然对数运算的 2 进制浮点数数据的软元件的起始编号	实数(2 进制)
D	保存运算结果的软元件起始编号	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

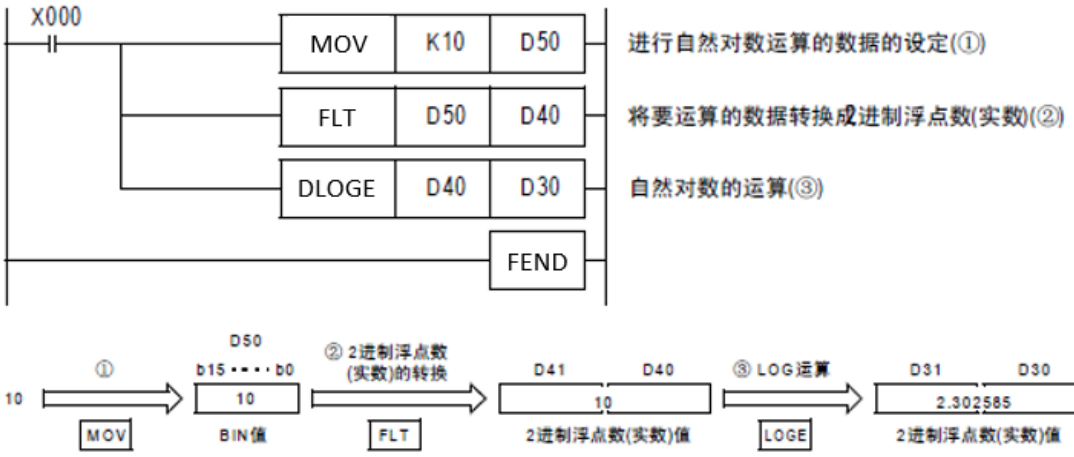
2、功能和动作说明

32 位运算指令，执行[S+1，S]的自然对数[以 e(2.17828) 为底时的对数] 运算,并将其运算结果保存到[D+1，D]中。此外，可以在 S 中直接指定实数。



3、程序举例

X000 为 ON 后，求出 D50 中设定的“10”的自然对数，并保存到 D30、D31 中的程序。



当 PLC 为 RUN 的时候进行自然对数$\log_e 15$的运算。

4、注意事项

（S+1，S）中指定的值只可以设定正数，指定为负数或 0 时报错 6706。

3.11.12 LOG10 指令(2 进制浮点数常用对数运算)

该指令执行常用对数运算。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DLOG10	DLOG10P	DLOG10 S D

操作数的数据类型如下表

操作数	内容	类型
S	保存进行常用对数运算的 2 进制浮点数数据的软	实数(2 进制)

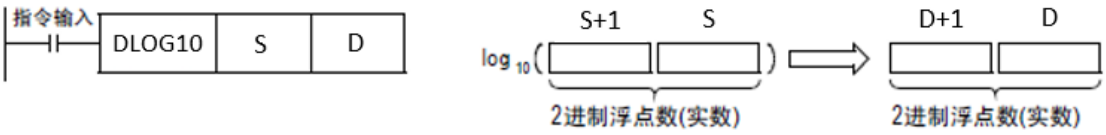
	元件的起始编号	
D	保存运算结果的软元件起始编号	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

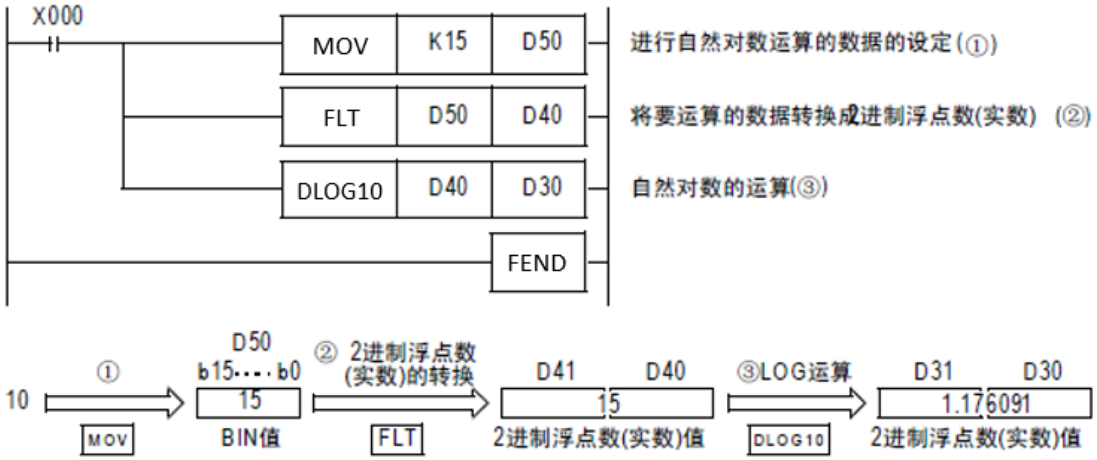
2、功能和动作说明

32 位运算指令，执行[S+1，S]的常用对数(10 为底时的对数)运算,并将其运算结果保存到[D+1，D]中。此外，可以在 S 中直接指定实数。



3、程序举例

X000 为 ON 后，求出 D50 中设定的“15”的常用对数，并保存到 D30、D31 中的程序。



当 PLC 为 RUN 的时候进行自然对数$\log_{10} 15$的运算。

4、注意事项

（S+1，S）中指定的值只可以设定正数，指定为负数或 0 时报错 6706。

3.11.13 ESQR 指令(2 进制浮点数开方运算)

2 进制浮点数开方(开根号)运算的指令。

1、指令格式见

16bit		32bit 9 步		指令格式
\	\	DESQR	DESQRP	DESQR S D

操作数的数据类型如下表

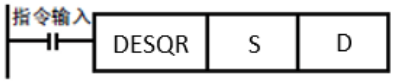
操作数	内容	类型
S	保存执行开方运算的 2 进制浮点数数据的软元件的起始编号	实数(2 进制)
D	保存运算结果的软元件起始编号	实数(2 进制)

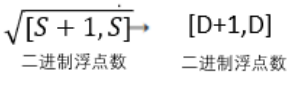
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

32 位运算指令，执行[S+1，S]进行开方(开根号)运算(2 进制浮点数运算)后,将其运算结果保存到[D+1，D]中。





3、程序举例

当 PLC 为 RUN 的时候对 S 进行开方。

4、注意事项

（1）（S+1，S）值为正数有效，为负数时报错 6706。

（2）运算结果为 0 时 M8020 置位。

软元件	名称	内容
M8020	零位	运算结果真为 0 时 ON

3.11.14 ENEG 指令(2 进制浮点数数据符号翻转)

使 2 进制浮点数(实数)数据的符号翻转的指令。

1、指令格式

16bit		32bit 5 步		指令格式
\	\	DENEG	DENEGP	DENEG D

操作数的数据类型如下表

操作数	内容	类型
D	保存要执行符号翻转的 2 进制浮点数数据的软元件的起始编号	实数(2 进制)

操作数的对象软元件如下表

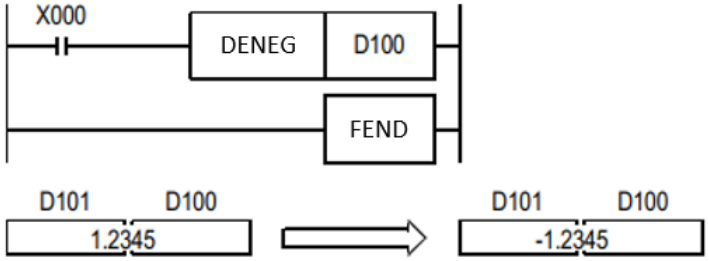
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D							●	●	●	●			●	

2、功能和动作说明

32 位运算指令，[D+1，D]的 2 进制浮点数数据的符号翻转，保存在[D+1，D]中。

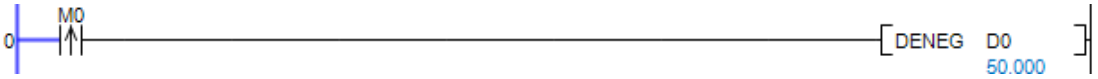
3、程序举例

X000 为 ON 时，将 D100、D101 的 2 进制浮点数数据的符号翻转，并且保存到 D100、D101 中的程序。

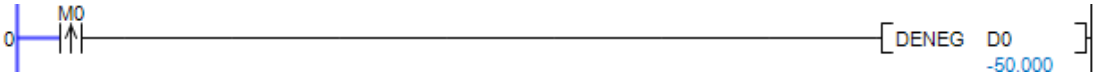


当 M0 为 ON 的时候，DENEG 对 D0 值的符号翻转。

翻转前



翻转后



4、注意事项

3.11.15 INT 指令(2 进制浮点数到 BIN 整数的转换)

将 2 进制浮点数，转换成可编程控制器中的一般数据形式 BIN 整数的指令(2 进制浮点数→BIN 整数)。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
INT	INTP	DINT	DINTP	INT S D

操作数的数据类型如下表

操作数	内容	类型
S	保存要转换成 BIN 整数的 2 进制浮点数数据的数据寄存器编号	实数(2 进制)
D	保存转换后的 BIN 整数的数据寄存器编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

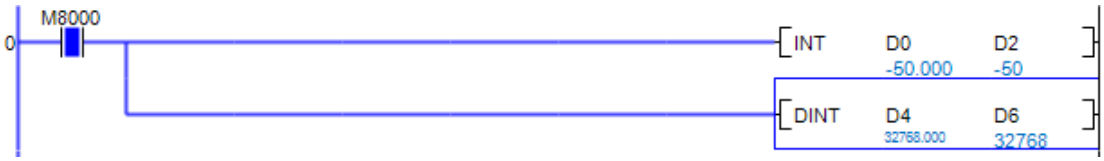
（1）16 位指令

[S+1，S]的 2 进制浮点数转换成 BIN 整数后，传送到 D 中。

（2）32 位指令

[S+1，S]的 2 进制浮点数转换成 BIN 整数后，传送到[D+1，D]中。

3、程序举例



4、注意事项

（1）相关标志位的动作

软元件	名称	内容
M8020	零位	运算结果真为 0 时 ON。
M8021	借位	位转换时，有因不满 1 而被舍去的情况发生时，置 ON。
M8022	进位	运算结果超出-32,768～32,767(16 位运算时)，或是

		-2,147,483,583～2,147,483,583(32 位运算时)的范围而产生溢出时为 ON。(运算结果不反映。)

（2）小数点以后的值被舍去

3.11.16 SIN 指令(2 进制浮点数正弦运算)

求角度(RAD)的 SIN 值的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DSIN	DSINP	DSIN S D

操作数的数据类型如下表

操作数	内容	类型
S	保存 2 进制浮点数的 RAD(角度)的软元件编号	实数(2 进制)
D	保存 2 进制浮点数的 SIN 值的软元件编号	实数(2 进制)

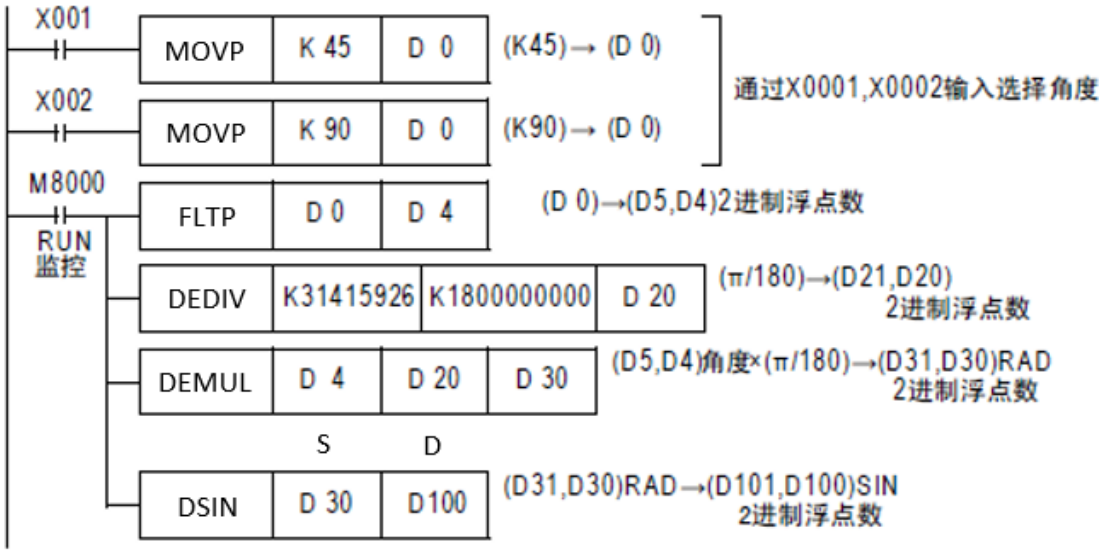
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

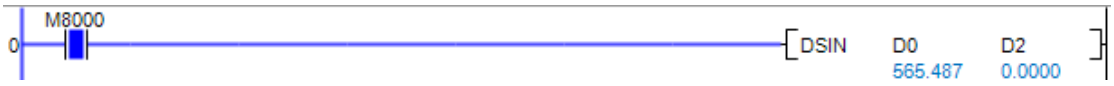
2、功能和动作说明

32 位运算指令,将[S+1，S]中指定的角度值(2 进制浮点数)转换成 SIN 值后，传送到[D+1，D]中。

3、程序举例



当 PLC 为 RUN 时，计算出 sin180° 的值



4、注意事项

运算结果为 0 时，M8020 置位。

3.11.17 COS 指令(2 进制浮点数余弦运算)

求角度(RAD)的 COS 值的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DCOS	DCOSP	DCOS S D

操作数的数据类型如下表

操作数	内容	类型
S	保存 2 进制浮点数的 RAD(角度)的软元件编号	实数(2 进制)
D	保存 2 进制浮点数的 COS 值的软元件编号	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

32 位运算指令,将[S+1，S]中指定的角度值(2 进制浮点数)转换成 COS 值后，传送到[D+1，D]中。

3、程序举例

当 PLC 为 RUN 的时候，计算 COS0 的值。

4、注意事项

无

3.11.18 TAN 指令(2 进制浮点数正切运算)

求角度(RAD)的 TAN 值的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DTAN	DTANP	DTAN S D

操作数的数据类型如下表

操作数	内容	类型
S	保存 2 进制浮点数的 RAD(角度)的软元件编号	实数(2 进制)
D	保存 2 进制浮点数的 TAN 值的软元件编号	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

32 位运算指令,将[S+1，S]中指定的角度值(2 进制浮点数)转换成 TAN 值后，传送到[D+1，D]中。

3、程序举例

当 PLC 为 RUN 时，计算 TAN90 的值

4、注意事项

运算结果为 0 时，M8020 置位。

3.11.19 ASIN 指令(2 进制浮点数反正弦运算)

执行 SIN^{-1} 运算的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DASIN	DASINP	DASIN S D

操作数的数据类型如下表

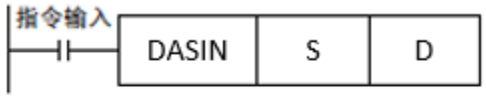
操作数	内容	类型
S	保存执行 SIN^{-1}(反正弦)运算的 SIN 值的软元件的起始编号	实数(2 进制)
D	保存运算结果的软元件起始编号	实数(2 进制)

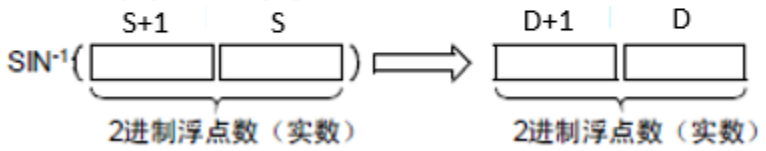
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

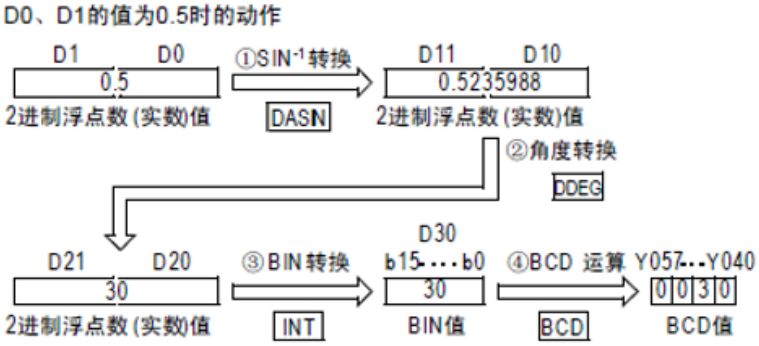
32 位运算指令,[S+1，S]的 SIN 值求出角度，将运算结果保存到[D+1，D]中。此外，可以在 S 中直接指定实数。





3、程序举例

X000 为ON时，求出D0、D1(2 进制浮点数) 的SIN-1，然后将其角度以BCD4位数形式输出到Y040 ～Y057中的程序。



当PLC为RUN时，求出D0、D1(2 进制浮点数) 的SIN-1，然后将其角度输出到D2、D3中的程序。

4、注意事项

（1）（S+1，S）的 SIN 值，设定在-1.0~1.0 范围，不在此范围报错 6706。

（2）（D+1，D）中保存的角度是弧度值（－π/2）~（π/2）。

3.11.20 ACOS 指令(2 进制浮点数反余弦运算)

执行 COS^{-1} 运算的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DACOS	DACOSP	DACOS S D

操作数的数据类型如下表

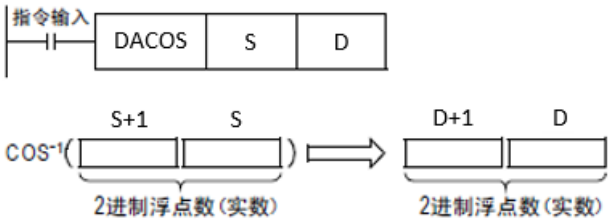
操作数	内容	类型
S	保存执行 COS^{-1}(反余弦)运算的 COS 值的软元件的起始编号	实数(2 进制)
D	保存运算结果的软元件起始编号	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

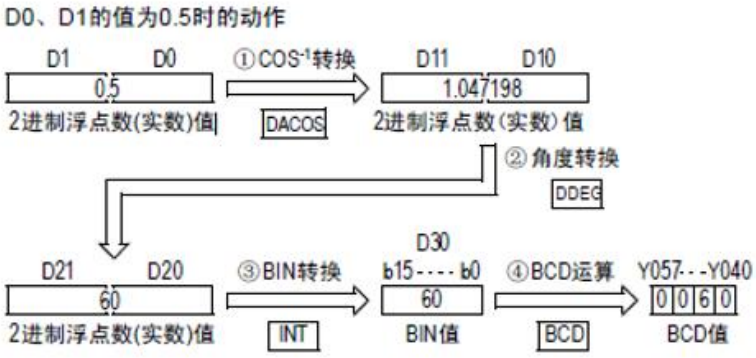
2、功能和动作说明

32 位运算指令,[S+1，S]的 COS 值求出角度，将运算结果保存到[D+1，D]中。此外，可以在 S 中直接指定实数。



3、程序举例

X000为ON时，求出D0、D1(2进制浮点数) 的COS-1，然后将其角度以BCD4位数形式输出到Y040～Y057中的程序。



PLC为RUN时，求出D0、D1(2进制浮点数) 的COS-1，然后将其角度输出到D2,D3中的程序。

4、注意事项

（1）（S+1，S）的 COS 值，设定在-1.0~1.0 范围，不在此范围报错 6706。

（2）（D+1，D）中保存的角度是弧度值 0~π。

3.11.21 ATAN 指令(2 进制浮点数反正切运算)

执行 TAN^{-1} 运算的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DATAN	DATANP	DATAN S D

操作数的数据类型如下表

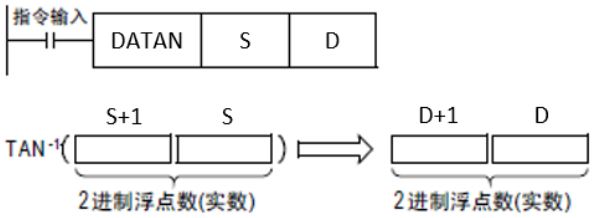
操作数	内容	类型
S	保存执行 TAN^{-1}(反正切)运算的 TAN 值的软元件的起始编号	实数(2 进制)
D	保存运算结果的软元件起始编号	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

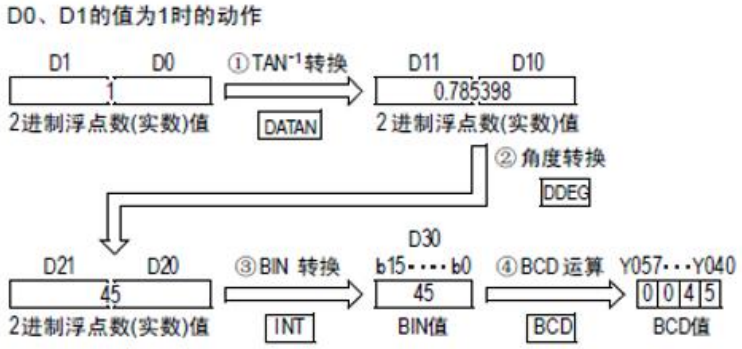
2、功能和动作说明

32 位运算指令,[S+1，S]的 TAN 值求出角度，将运算结果保存到[D+1，D]中。此外，可以在 S 中直接指定实数。



3、程序举例

X000为ON时，求出D0、D1(2 进制浮点数) 的TAN-1，然后将其角度以BCD4位数形式输出到Y040～Y057中的程序。



当PLC为RUN时，求出D0、D1(2 进制浮点数) 的TAN-1，然后将其角度输出到D2、D3中的程序。

4、注意事项

（D+1，D）中保存的角度是弧度值>(-π/2)或<(π/2)

关于弧度与角度之间的转换，请参考RAD指令、DEG指令。

→ 有关RAD指令，请参考2.13.22节

→ 有关 DEG 指令，请参考 2.13.23 节

3.11.22 RAD 指令(2 进制浮点数角度到弧度的转换)

将角度单位的值转换成弧度单位的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DRAD	DRADP	DRAD S D

操作数的数据类型如下表

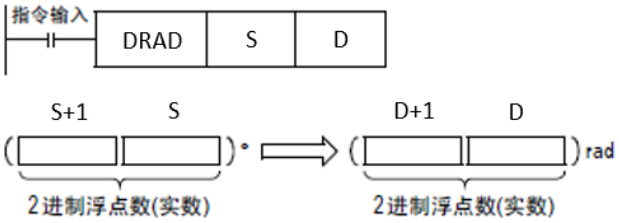
操作数	内容	类型
S	保存要转换成弧度单位的角度的软元件起始编号	实数(2 进制)
D	保存运算结果的软元件起始编号	实数(2 进制)

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S													●	
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

32 位运算指令,[S+1，S]的单位从角度单位转换成弧度单位 ，将运算结果保存到[D+1，D]中。此外，可以在 S 中直接指定实数。

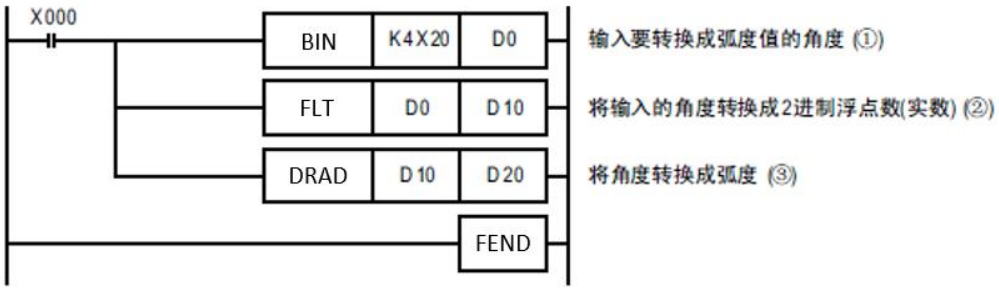


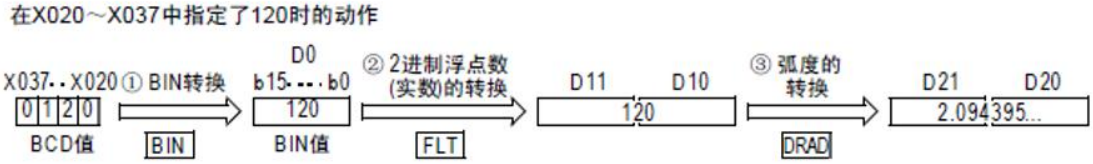
角度单位→弧度单位的转换如下所示执行。

弧度单位＝角度单位× $\frac{\pi}{180}$

3、程序举例

X000为ON时，将X020～X037中以BCD4位数形式设定的角度转换成弧度，以2进制浮点数形式保存在D20、D21中的程序。





将 D0 保存的弧度单位变成 D10 的角度单位

4、注意事项

无

3.11.23 DEG 指令(2 进制浮点数弧度到角度的转换)

将弧度单位的值转换成角度(DEG)单位的指令。

1、指令格式

16bit		32bit 9 步		指令格式
\	\	DDEG	DDEGP	DDEG S D

操作数的数据类型如下表

操作数	内容	类型
S	保存要转换成角度单位的弧度的软元件起始编号	实数(2 进制)
D	保存运算结果的软元件起始编号	实数(2 进制)

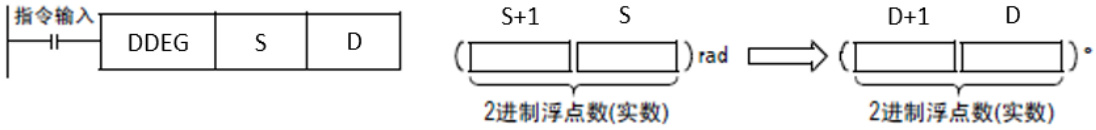
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
S													●	
D														

操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●	●	●			●	
D							●	●	●	●			●	

2、功能和动作说明

32 位运算指令,[S+1，S]的单位从弧度单位转换成角度单位 ，将运算结果保存到[D+1，D]中。

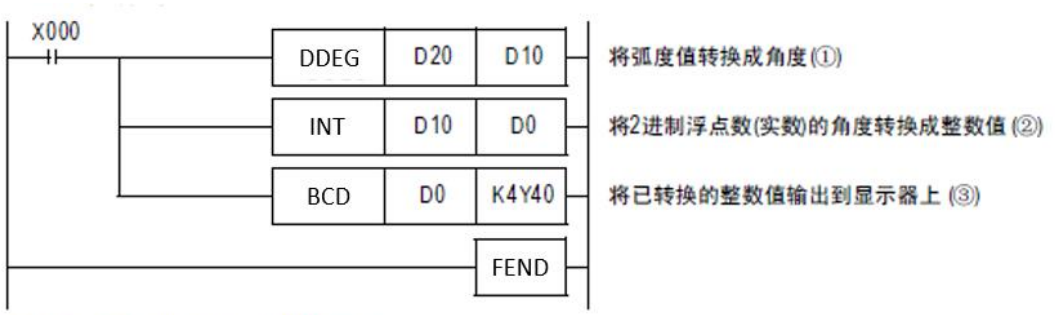


弧度单位→角度单位的转换如下所示执行。

角度单位＝弧度单位× $\frac{180}{\pi}$

3、程序举例

当X000为ON时，将D20、D21中以2进制浮点数形式设定的弧度值转换成角度后，以BCD值的形式输出到Y040～Y057 中的程序。



D20、D21的值为1.435792时的动作

将 D0 保存的角度单位变成 D10 的弧度单位。

4、注意事项

无

3.12 数据处理 2 指令

3.12.1 WSUM 指令(算出数据合计值)

该指令可计算出连续的 16 位或是 32 位数据的合计值。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
WSUM	WSUMP	DWSUM	DWSUMP	WSUM S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存要算出合计值的数据的软元件起始编号	BIN16/32 位
D	保存合计值的软元件起始编号	BIN32/64 位
n	数据个数(0＜n)	BIN16/32 位

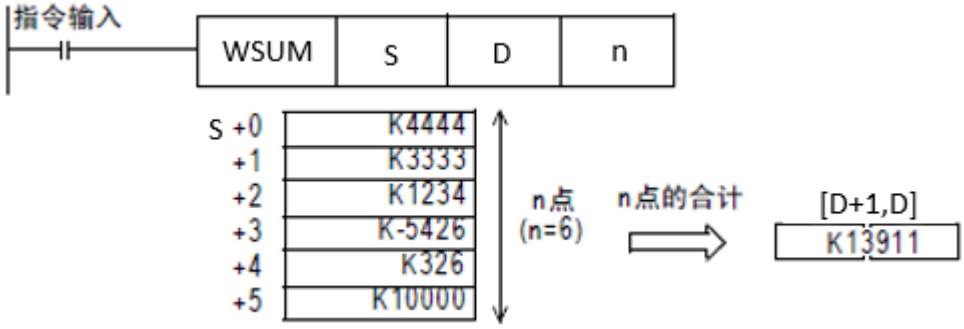
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●			●	

2、功能和动作说明

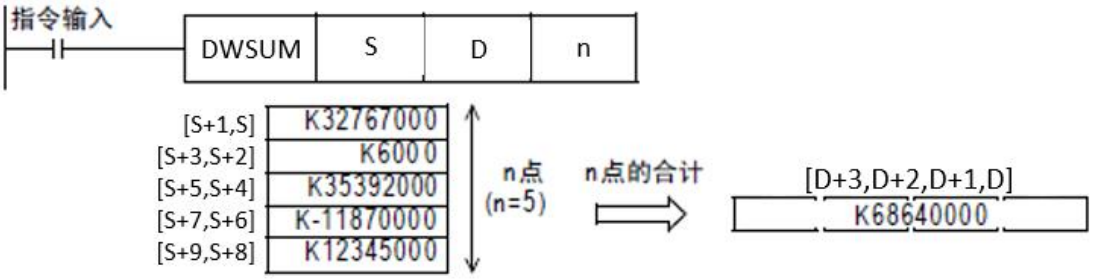
（1）16 位指令

将 S 开始的 n 点 16 位数据的合计值，以 32 位数据形式保存在[D+1，D]中。



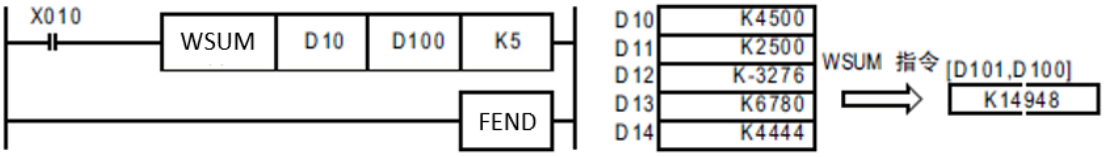
（2）32 位指令

将（S+1，S）开始的 n 点 32 位数据的合计值，以 64 位数据形式保存在（D+3，D+2，D+1，D）中。

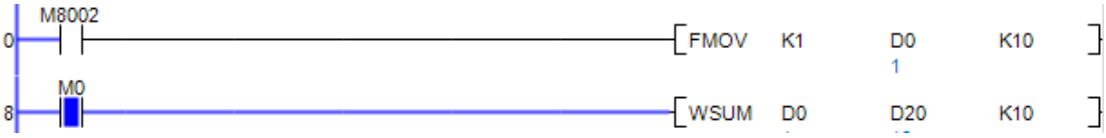


3、程序举例

当 X010 为 ON 时，将 D10～D14 的 16 位数据合计值保存到[D101，D100]中的程序。



当 M0 为 ON 时，将 D0~D9 的数据累加起来放到 D20 里去。



4、注意事项

（1）数据个数设定≤0 时，报错 6706。

（2）注意运算结果，16 位指令时，超过 16 位数的情况下，查看 D+1，D 分别是运算结果的高 16 位、低 16 位，未超过查看 D 即可；32 位指令时，超过的情况下，查看（D+3，D+2），（D+1，D）分别是运算结果的高 32 位、低 32 位，未超过时查看（D+1,D ）即可。

3.12.6 SWAP 指令(高低字节互换)

互换字数据的高 8 位和低 8 位的指令。

1、指令格式

16bit 3 步		32bit 5 步		指令格式
SWAP	SWAPP	DSWAP	DSWAPP	SWAP S

操作数的数据类型如下表

操作数	内容	类型
S	高低字节互换的字软元件	BIN16 位/32 位

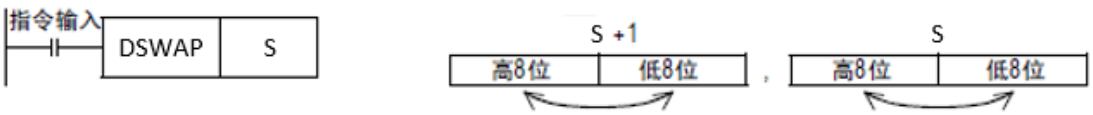
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

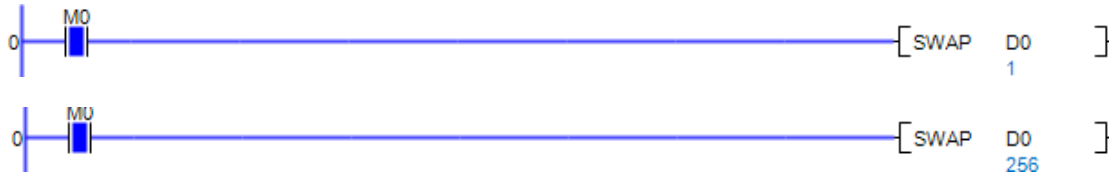
（1）16 位指令

（2）32 位指令

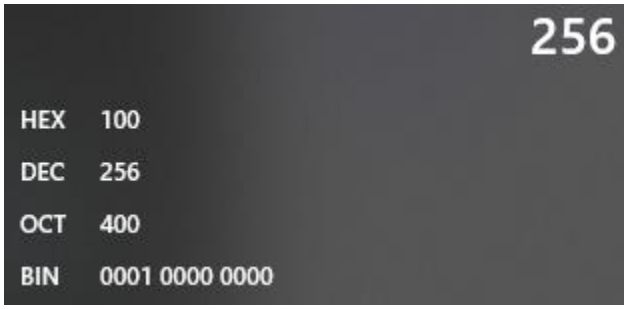


3、程序举例

M0 为 ON 时，将 D0 的低 8 位与高 8 位互换。



1 的二进制为 0000 0000 0000 0001



256 的二进制为 0000 0001 0000 0000

4、注意事项

即使是 32 位指令，也是执行各自的低 8 位和高 8 位的互换。与 XCH 指令的扩展功能相同。

3.12.2 WTOB 指令(字节单位的数据分离)

将连续的 16 位数据按照字节(8 位)单位进行分离的指令。

1、指令格式

16bit 7 步		32bit		指令格式
WTOB	WTOBP	\	\	WTOB S D n

操作数的数据类型如下表

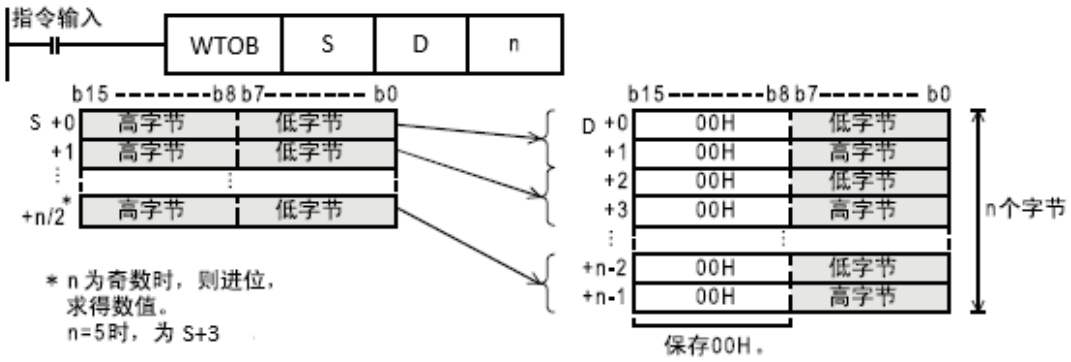
操作数	内容	类型
S	保存要按照字节单位进行分离的数据的软元件起始编号	BIN16 位
D	保存已经按照字节单位分离的结果的软元件起始编号	BIN16 位
n	要分离的字节数据个数(0≦n)	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●			●	

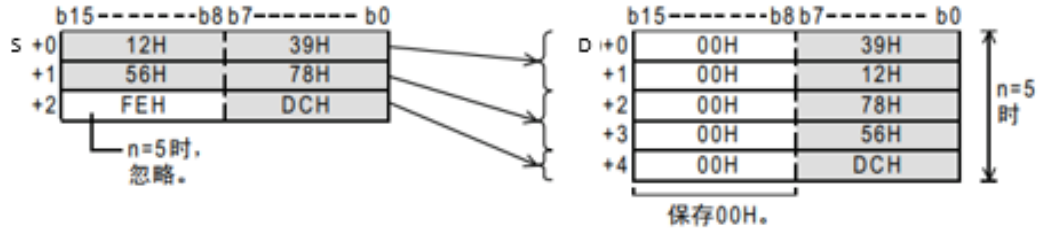
2、功能和动作说明

（1）16 位指令，将 S 开始的 n/2 个软元件中保存的 16 位数据分离成 n 个字节，如下所示保存到以 D 开始的 n 点软元件中。



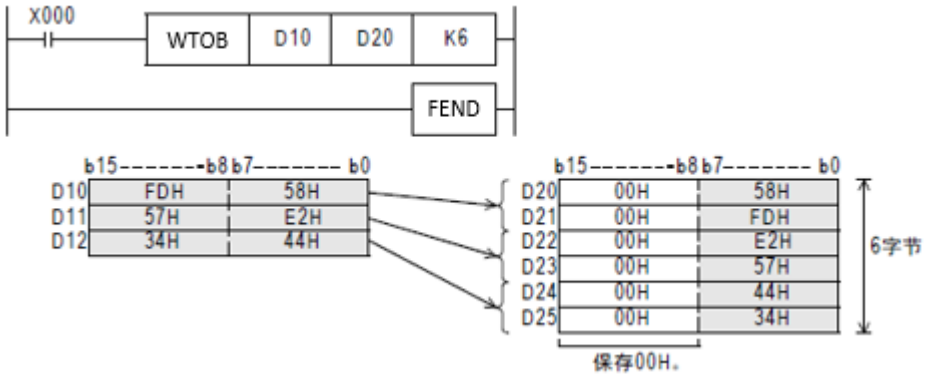
（2）在保存分离后字节数据的软元件(D 以后)的高字节(8 位)中，保存 00H。

（3）n 为奇数时，如下图所示，在分离源的最终数据中，只有低字节(8 位)为对象数据。

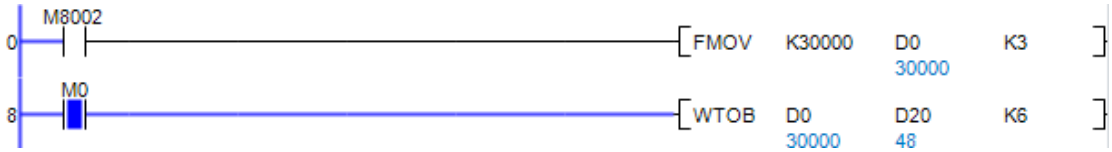


3、程序举例

当 X000 为 ON 后，将 D10～D12 的数据按照字节单位分离，然后保存到 D20～D25 中的程序。



当 M0 为 ON 时，将 D0~D2 的数据按照字节单位分离，然后保存到 D20~D25 中的程序。



		D20	48
		D21	117
		D22	48
		D23	117
D0	30000	D24	48
D1	30000	D25	117
D2	30000	D26	0

4、注意事项

（1）当分离源软元件的 S～(S+n/2)超出了指定软元件的软元件范围时。 n 为奇数时，需要占用进位后数值的个数部分的软元件，报错 6706。

（2）当保存已分离的数据的软元件 D～(D+n-1)超出了指定软元件的软元件范围时，报错 6706。

（3）n=0 时，不执行指令。

3.12.3 BTOW 指令(字节单位的数据结合)

将连续的 16 位数据的低 8 位(低字节)结合在一起的指令。

1、指令格式

16bit 7 步		32bit		指令格式
BTOW	BTOWP	\	\	BTOW S D n

操作数的数据类型如下表

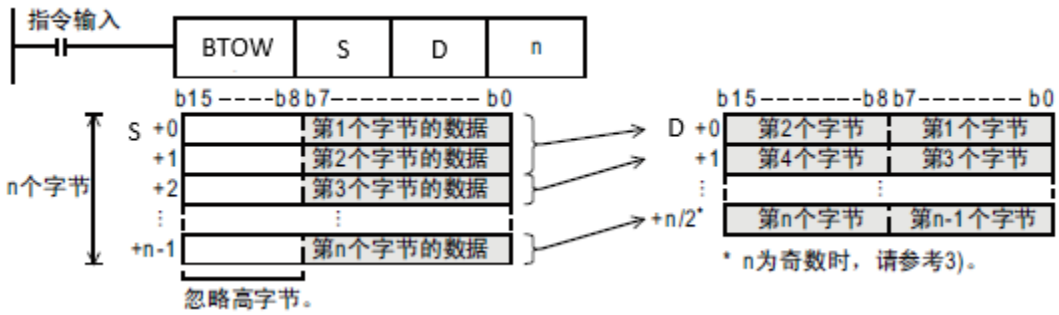
操作数	内容	类型
S	保存要按照字节单位结合的数据的软元件起始编号	BIN16 位
D	保存已经按照字节单位结合的结果的软元件起始编号	BIN16 位
n	要结合的字节数据个数(0≦n)	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●			●	

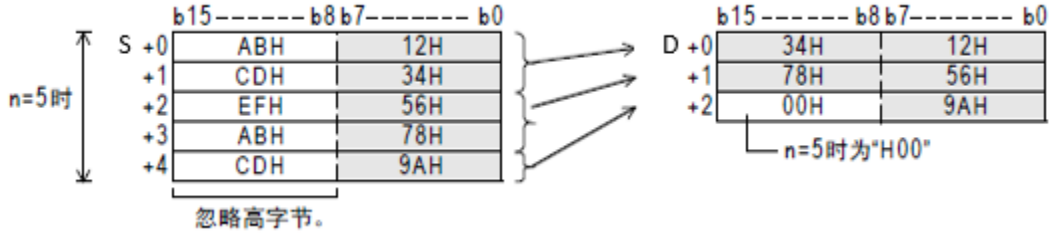
2、功能和动作说明

（1）16 位指令，将 S 开始的 n 点 16 位数据的低字节(8 位)结合在一起后的 16 位数据，如下所示保存到以 D 开始的 n/2 点软元件中。



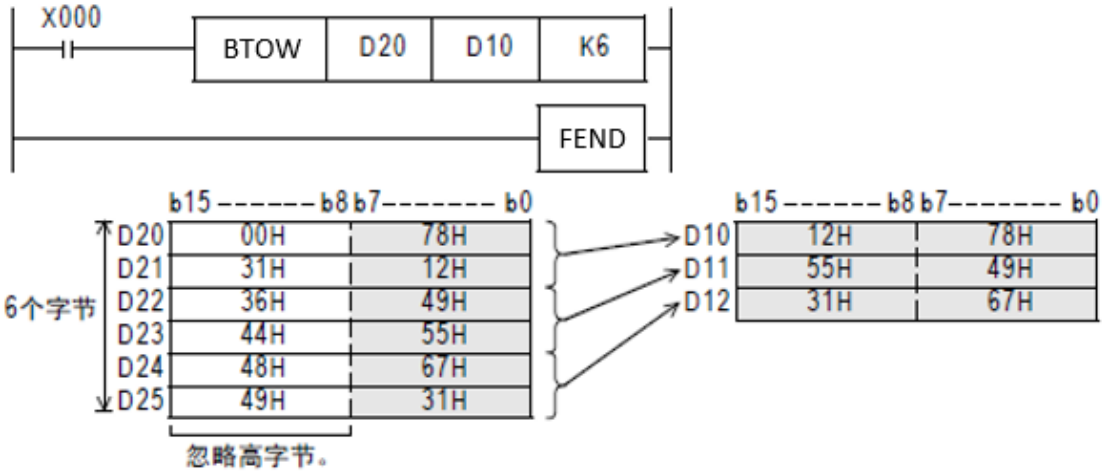
（2）结合源的 16 位数据(S 以后)的高字节(8 位)被忽略。

（3）n 为奇数时，如下图所示，最终结合后的数据的高字节(8 位)为 00H。

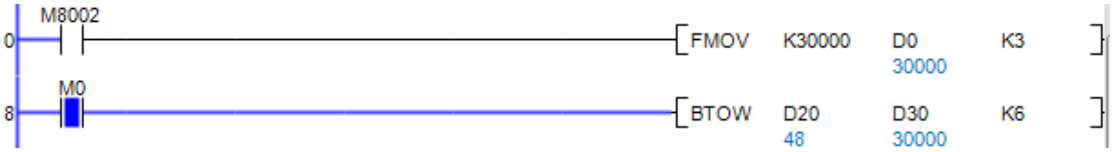


3、程序举例

当 X000 为 ON 后，D20～D25 的低字节(8 位)数据被结合后，保存到 D10～D12 中的程序。



当 M0 为 ON 时，D20～D25 的低字节(8 位)数据被结合后，保存到 D30～D32 中的程序。



D20	48
D21	117
D22	48
D23	117
D24	48
D25	117

D30	30000
D31	30000
D32	30000

4、注意事项

（1）结合源的软元件的 S～(S+n-1)中指定的软元件超出了该软元件范围时，报错 6706。

（2）当保存已结合数据的软元件 D～(D +n/2) 超出了指定软元件的软元件范围时。当 n 为奇数时，需要占用进位后数值的个数部分的软元件，报错 6706。

（3）n=0 时，不执行指令。

3.12.4 UNI 指令(16 位数据的 4 位结合)

将连续的 16 位数据的低 4 位结合在一起的指令。

1、指令格式

16bit 7 步		32bit		指令格式
UNI	UNIP	\	\	UNI S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存要结合的数据的软元件起始编号	BIN16 位
D	保存已结合的数据的软元件编号	BIN16 位
n	结合数(0～4，n=0 时不处理)	BIN16 位

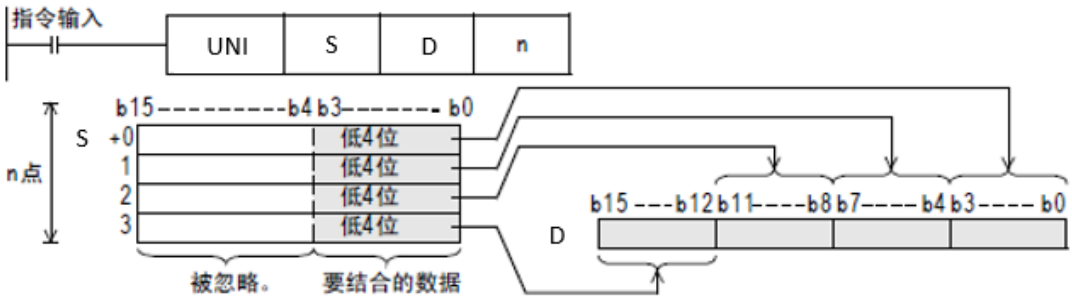
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
S														
D														

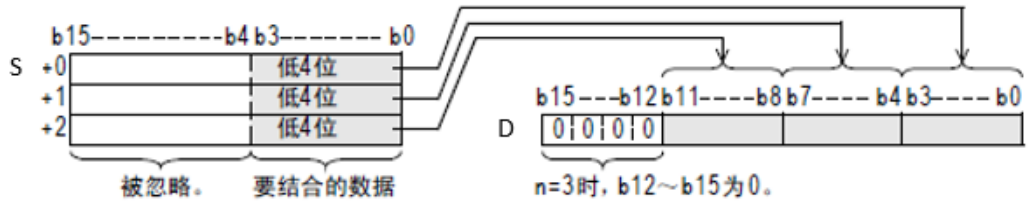
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●			●	

2、功能和动作说明

16 位指令，将 S 开始的 n 点 16 位数据的低 4 位结合后的 16 位数据，如下所示保存到 D 中。

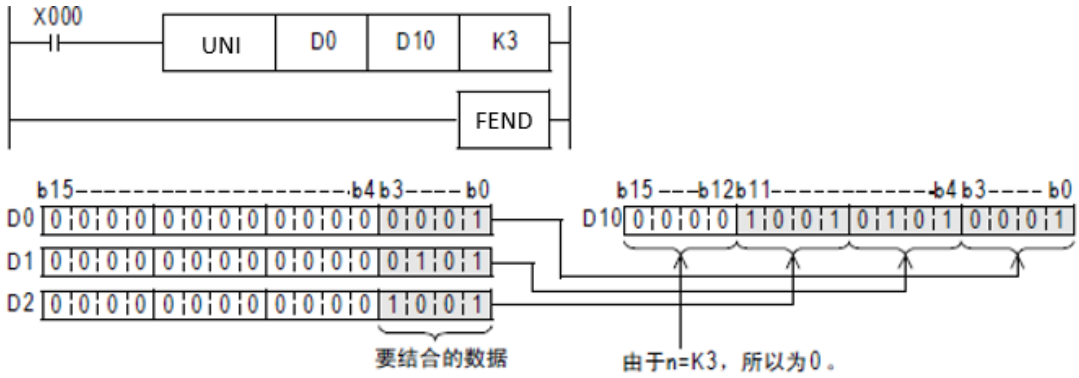


1≦n≦3时， D的高位{4×(4-n)}个位为0。
例如，n=3 时， S～(S+2)的低 4 位被保存到 D 的 b0～b11 中， D 的高 4 位变为 0。



3、程序举例

X000 为 ON 后，将 D0~D2 的低 4 位结合后，保存到 D10 中。



M0 为 ON 后，将 D0~D2 的低 4 位结合后，保存到 D10 中。

4、注意事项

以下一些情况下会发生运算错误，错误标志位M8067置ON，错误代码保存在D8067中。
S～(S+ n)中指定的软元件超出了该软元件范围时。(错误代码：K6706)
n指定了0～4以外的数字时。(错误代码：K6706)
n=0 时，不执行指令。

3.12.5 DIS 指令(16 位数据的 4 位分离)

将连续的 16 位数据的低 4 位结合在一起的指令。

1、指令格式

16bit 7 步		32bit		指令格式
DIS	DISP	\	\	DIS S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存要分离的数据的软元件起始编号	BIN16 位
D	保存已分离的数据的软元件编号	BIN16 位
n	分离数(0～4，n=0 时不处理)	BIN16 位

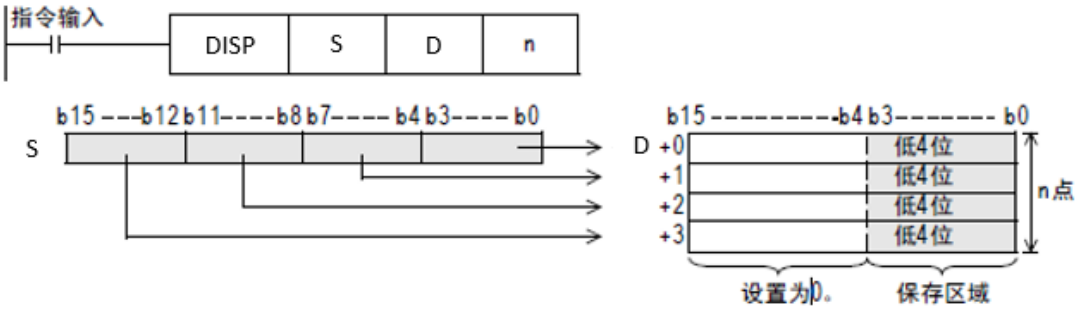
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”

S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●			●	

2、功能和动作说明

（1）16 位指令，将 S 开始的 16 位数据以 4 位结合后为单位分离，如下所示保存到 D 中。

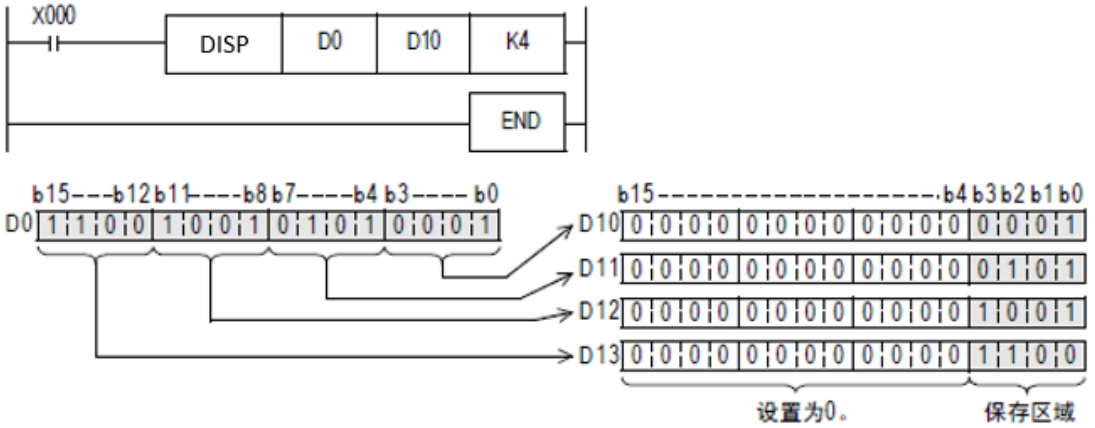


（2）n 范围(0～4，n=0 时不处理)

（3）D 开始的 n 点软元件的高 12 位设置为 0。

3、程序举例

当 X000 为 ON 后，将 D0 每隔 4 个位分离后，保存到 D10～D13 中的程序。



当 M0 为 ON 后，将 D10 每隔 3 个位分离后，保存到 D50～D52 中的程序。

D50	1
D51	5
D52	9

4、注意事项

（1）当 D 开始的 n 点软元件超出了知道软元件的范围时，报错 6706。

（2）n 指定的 0~4 意外数字报错 6706。

3.12.7 SORT2 指令(数据排序 2)

以指定的群数据(列)为基准，以行为单位，将由数据(行)和群数据(列)构成的数据表进行升序/降序重新排序的指令。在这个指令中，由于是在连续的软元件中保存数据(行方向)，所以便于增加数据(行)。

1、指令格式

16bit 11 步		32bit 21 步		指令格式
SORT2	\	DSORT2	\	SORT2 S m1 m2 D n

操作数的数据类型如下表

操作数	内容	类型
S	保存数据表格的软元件起始编号[占用 m1×m2 点]	BIN16 位/32 位
m1	数据(行)数[1～32]	BIN16 位/32 位
m2	群数据(列)数[1～6]	BIN16 位/32 位
D	保存运算结果的软元件起始编号[占用 m1×m2 点]	BIN16 位/32 位
n	作为排序标准的群数据(列)的列编号[1～m2]	BIN16 位/32 位

操作数的对象软元件如下表

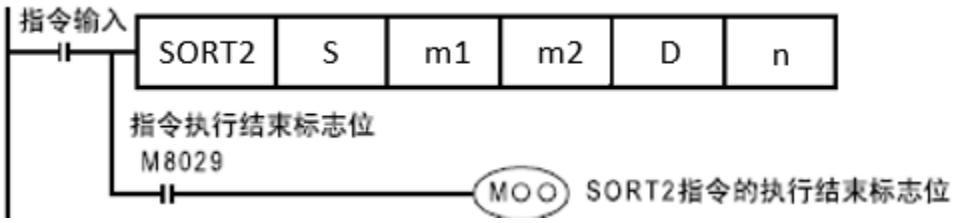
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
S2											●	●		
S3											●	●		

S														
D											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1							●	●	●	●				
S2							●	●	●	●				
S3														
S							●	●	●	●				
D							●	●	●	●				

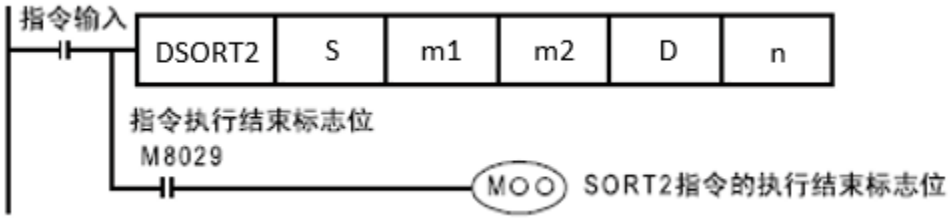
2、功能和动作说明

（1）16 位指令

针对 开始的(m1×m2)点的数据表格(排序前)，以 n 列的群数据为基准，将数据行进行升序或是降序的排列，然后保存到从 开始的(m1×m2)点的数据表格(排序后)中。



（2）32 位指令



针对(S+1,S)开始的(m1×m2)点的数据表格(排序前)，以 n 列的群数据为基准，将数据行进行升序或是降序的排列，然后保存到从(D+1,D)开始的(m1×m2)点的数据表格(排序后)中。

3、程序举例

在“n=K2(列号 2)”和“n=K3(列号 3)”的情况下，对如下所示的排序前的数据进行排序，动作如下所示。下面列举了 16 位运算的动作例子。执行 32 位运算时，请使用 BIN32 位构成数据表格。此外，如果先在第 1 列中输入管理编号等连续编号，则可以根据其内容判断出原来所在的行号，因此非常方便。

排序前数据：

行号	列号	群数 m2 个（m2=K4 时）			
		1	2	3	4
		管理编号	身高	体重	年龄
数据数 m1=K5 的情况	1	S	S+1	S+2	S+3
		1	150	45	20
	2	S+4	S+5	S+6	S+7
		2	180	50	40
	3	S+8	S+9	S+10	S+11
		3	160	70	30
	4	S+12	S+13	S+14	S+15
		4	100	20	8
	5	S+16	S+17	S+18	S+19
		5	150	50	45

（1）以 n=K2(列号 2)为基准执行指令时的排序结果(升序的情况)

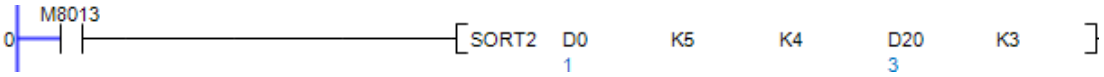
行号	列号	群数 m2 个（m2=K4 时）			
		1	2	3	4
		管理编号	身高	体重	年龄
数据数 m1=K5 的情况	1	D	D+1	D+2	D+3
		4	100	20	8
	2	D+4	D+5	D+6	D+7
		1	150	45	20
	3	D+8	D+9	D+10	D+11
		5	150	50	45
	4	D+12	D+13	D+14	D+15

		3	160	70	30
	5	D+16	D+17	D+18	D+19
		2	180	50	40

（2）以 n=K3(列号 3)为基准执行指令时的排序结果(降序的情况)

列号 / 行号		群数 m2 个（m2=K4 时）			
		1	2	3	4
		管理编号	身高	体重	年龄
数据数 m1=K5 的情况	1	D	D+1	D+2	D+3
		3	160	70	30
	2	D+4	D+5	D+6	D+7
		2	180	50	40
	3	D+8	D+9	D+10	D+11
		5	150	50	45
	4	D+12	D+13	D+14	D+15
		1	150	45	20
	5	D+16	D+17	D+18	D+19
		4	100	20	8

当 M8013 为 ON 时，将对表格中 3 列的数据进行排序，升降序由 M8165 决定。



元件	+0	+1	+2	+3	+4	+5	+6	+7
D0	1	150	45	20	2	180	50	40
D8	3	160	70	30	4	100	20	8
D16	5	150	50	45	3	160	70	30
D24	2	180	50	40	5	150	50	45
D32	1	150	45	20	4	100	20	8

M8165 为 ON 根据 K3 列的数据排序（降序）的结果可查看 D20~D39 部分对应上表。

4、注意事项

（1）相关软元件的动作

M8029：指令结束标志，排序结束时置位；

M8165：置 ON 时降序排列，OFF 时升序排列。

（2）以下操作会引起错误排序，请注意

- 动作过程中，请勿使操作数和数据的内容变化。
- 再次执行时，请将指令输入 OFF 一次。
- 指令的使用次数的限制在程序中最多可同时驱动 2 次。
- 原来的数据和排序替换后的数据，请错开，不要重叠。

3.13 时钟运算指令

3.13.1 TCMP 指令(时钟数据比较)

将比较基准时间和时间数据进行大小比较，根据比较的结果控制位软元件 ON/OFF。。

1、指令格式

16bit 11 步		32bit		指令格式
TCMP	TCMPP	\	\	TCMP S1 S2 S3 S D

操作数的数据类型如下表

操作数	内容	类型
S1	指定比较基准时间的“时”。[设定范围：0～23]	BIN16 位
S2	指定比较基准时间的“分”。[设定范围：0～59]	BIN16 位
S3	指定比较基准时间的“秒”。[设定范围：0～59]	BIN16 位
S	指定时间数据(时、分、秒)的“时”。(占用 3 点)	BIN16 位
D	根据比较结果 ON/OFF 位软元件。(占用 3 点)	位

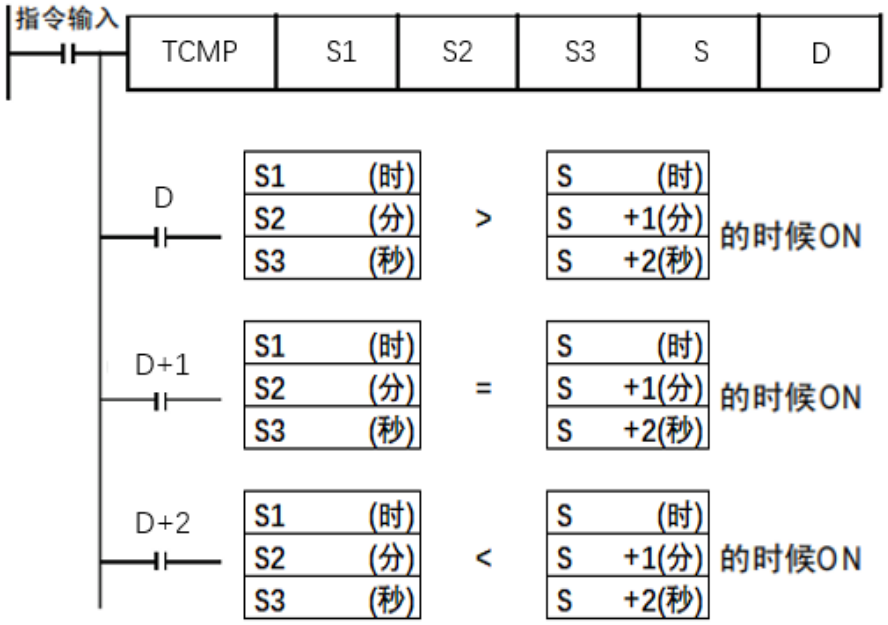
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		

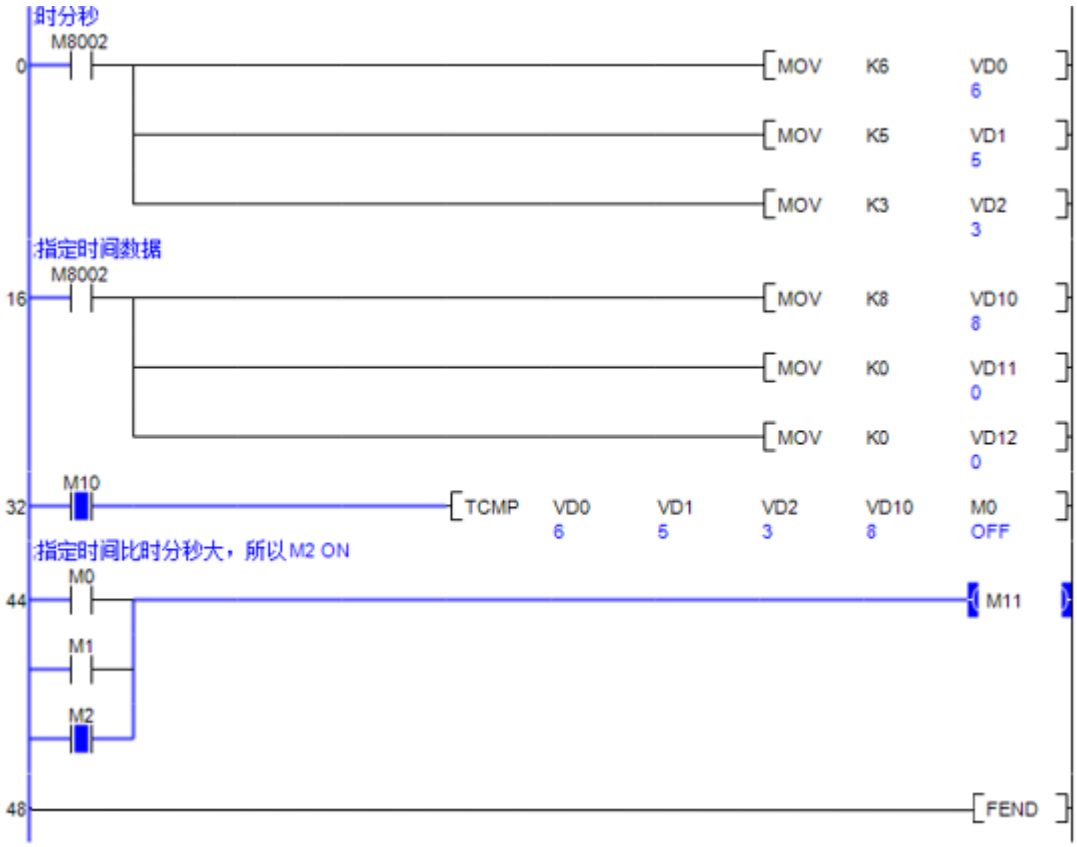
S3											●	●		
S														
D		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
S3	●	●	●	●	●	●	●	●	●	●	●	●	●	
S					●	●	●	●	●	●			●	
D													●	

2、功能和动作说明

将比较基准时间(时、分、秒)[S1，S2 ，S3] 的时间与时间数据(时、分、秒)[S ， S+1，S+2]进行大小比较，根据其大小一致的结果 ON/OFF D 开始的 3 点。



3、程序举例



4、注意事项

（1）即使是指令输入为 OFF,TCMP 指令不执行时，D ～D +2 也会保持当指令输入从 ON 变为 OFF 之前的状态。

（2）使用可编程控制器内置实时时钟的时钟数据的时间(时、分、秒)时，请使用 TRD 指令，读出特殊数据寄存器的值以后，在各个操作数中指定其字软元件。

3.13.2 TZCP 指令(时钟数据区间比较)

将上下 2 点的比较基准时间和时间数据进行大小比较,根据比较的结果控制指定位软元件的 ON/OFF。

1、指令格式

16bit 9 步		32bit		指令格式
TZCP	TZCPP	\	\	TZCP S1 S2 S3 D

操作数的数据类型如下表

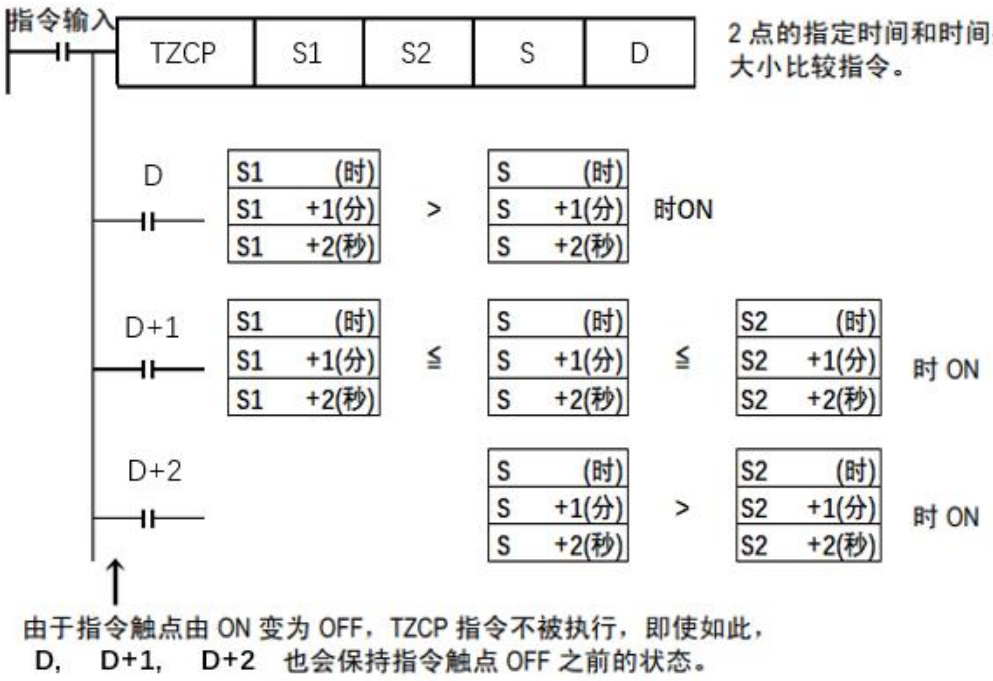
操作数	内容	类型
S1	指定比较下限时间(时、分、秒)的“时”。(占用 3 点)	BIN16 位
S2	指定比较上限时间(时、分、秒)的“时”。(占用 3 点)	BIN16 位
S3	指定时间数据(时、分、秒)的“时”。(占用 3 点)	BIN16 位
D	根据比较结果 ON/OFF 位软元件。(占用 3 点)	位

操作数的对象软元件如下表

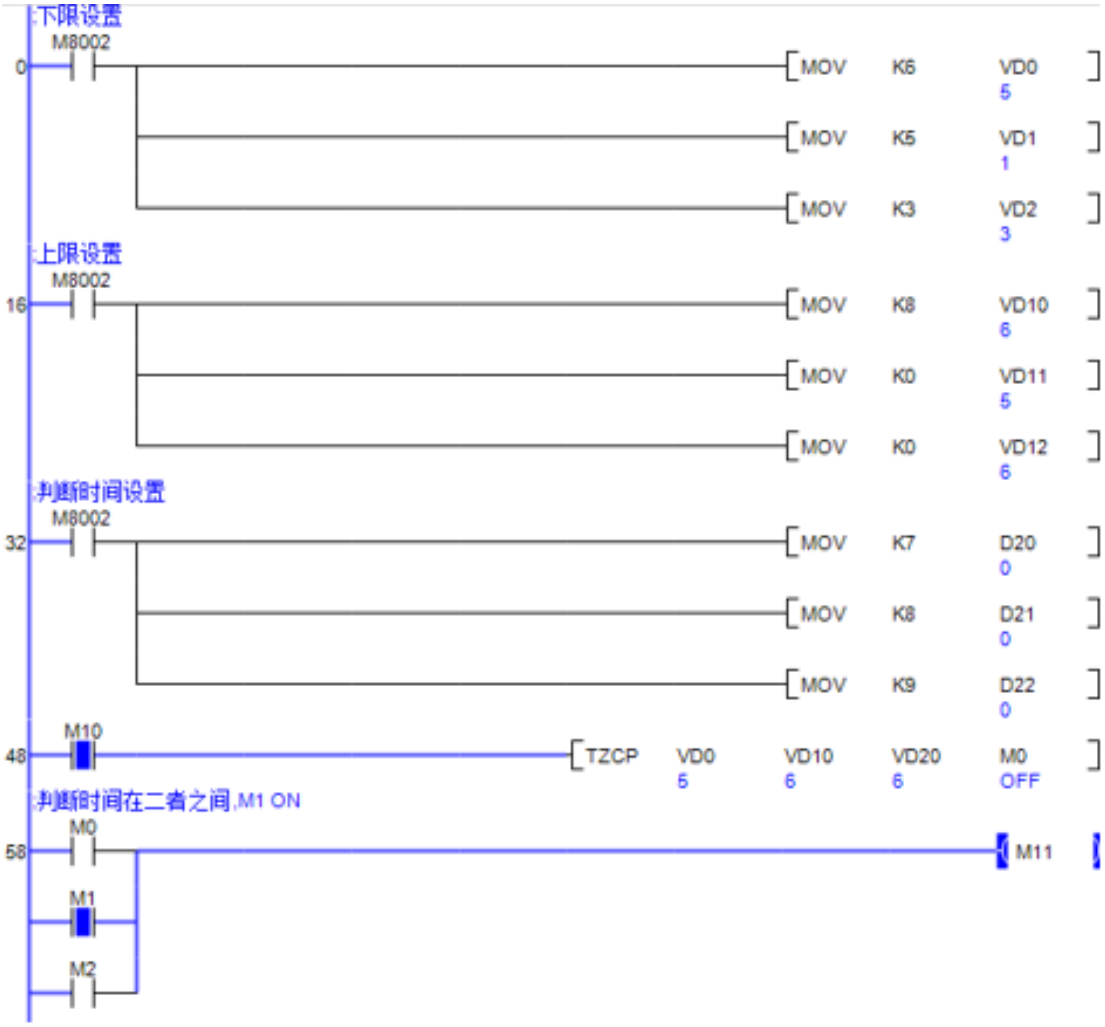
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
S2														
S3														
D		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1					●	●	●	●	●	●			●	
S2					●	●	●	●	●	●			●	
S3					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	

2、功能和动作说明

将上下 2 点比较基准时间(时、分、秒)与以 S 开头的 3 点时间数据(时、分、秒)进行比较，根据比较的结果 ON/OFF 从 D 开始的 3 点软元件。


指令输入
TZCP
S1
S2
S
D
2 点的指定时间和时间数据的大小比较指令。
D
S1 (时)
S1 +1(分)
S1 +2(秒)
>
S (时)
S +1(分)
S +2(秒)
时ON
D+1
S1 (时)
S1 +1(分)
S1 +2(秒)
≦
S (时)
S +1(分)
S +2(秒)
≦
S2 (时)
S2 +1(分)
S2 +2(秒)
时 ON
D+2
S (时)
S +1(分)
S +2(秒)
>
S2 (时)
S2 +1(分)
S2 +2(秒)
时 ON
由于指令触点由 ON 变为 OFF，TZCP 指令不被执行，即使如此，
D， D+1， D+2 也会保持指令触点 OFF 之前的状态。

3、程序举例



4、注意事项

用可编程控制器内置实时时钟的时钟数据的时间(时、分、秒)时，请使用 TRD 指令，读出特殊数据寄存器的值以后，在各个操作数中指定其字软元件。

3.13.3 TADD 指令(时钟数据加法运算)

将 2 个时间数据进行加法运算后，保存在字软元件中。

1、指令格式

16bit 7 步		32bit		指令格式
TADD	TADDP	\	\	TADD S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	指定进行加法运算的时间数据(时、分、秒)的“时”。(占用 3 点)	BIN16 位
S2	指定进行加法运算的时间数据(时、分、秒)的“时”。(占用 3 点)	BIN16 位
D	保存两个时间数据(时、分、秒)加法运算的结果。(占用 3 点)	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
S2														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1					●	●	●	●	●	●			●	
S2					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	

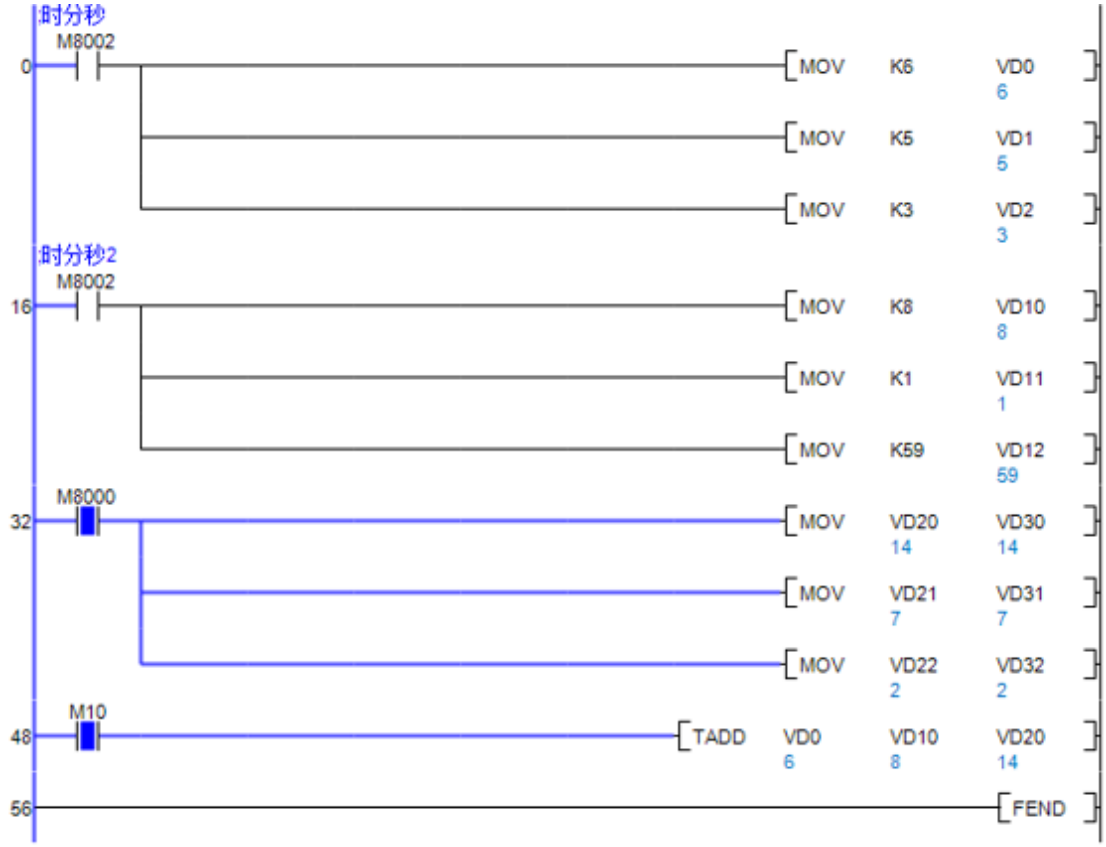
2、功能和动作说明

将[S1，S1+1，S1+2] 的时间数据(时、分、秒) 与[S2，S2+1，S2+2] 的时间数据(时、分、秒)进行加法运算，其结果保存到[D，D+1，D+2](时、分、秒)中。

(S1,S1+1,S1+2)+(S2,S2+1,S2+2)
➜(D,D+1,D+2)

S1	(时)	+	S2	(时)	➜	D	(时)	“时”的范围为[0~23]
S1	+1(分)		S2	+1(分)		D	+1(分)	“分”的范围为[0~59]
S1	+2(秒)		S2	+2(秒)		D	+2(秒)	“秒”的范围为[0~59]

3、程序举例



4、注意事项

主要相关标志位的动作：当运算结果超出 24 小时时，进位标志位变为 ON，从单纯的加法运算值中减去 24 个小时后将该时间作为运算。运算结果为 0(0 时 0 分 0 秒)时，零位标志位变为 ON。

3.13.4 TSUB 指令(时钟数据减法运算)

将 2 个时间数据进行减法运算后，保存在字软元件中。

1、指令格式

16bit 7 步		32bit		指令格式
TSUB	TSUBP			TSUB S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	指定进行减法运算的时间数据(时、分、秒)的“时”。(占用 3 点)	BIN16 位

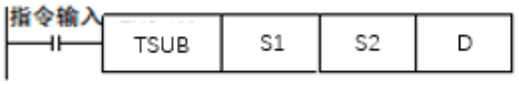
S2	指定进行减法运算的时间数据(时、分、秒)的“时”。(占用 3 点)	BIN16 位
D	保存两个时间数据(时、分、秒)减法运算的结果。(占用 3 点)	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
S2														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1					●	●	●	●	●	●			●	
S2					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	

2、功能和动作说明

从[S1，S1+1，S1+2] 的时间数据(时、分、秒) 中减去[S2，S2+1，S2+2] 的时间数据(时、分、秒)，其结果保存到[D，D+1，D+2](时、分、秒)中。

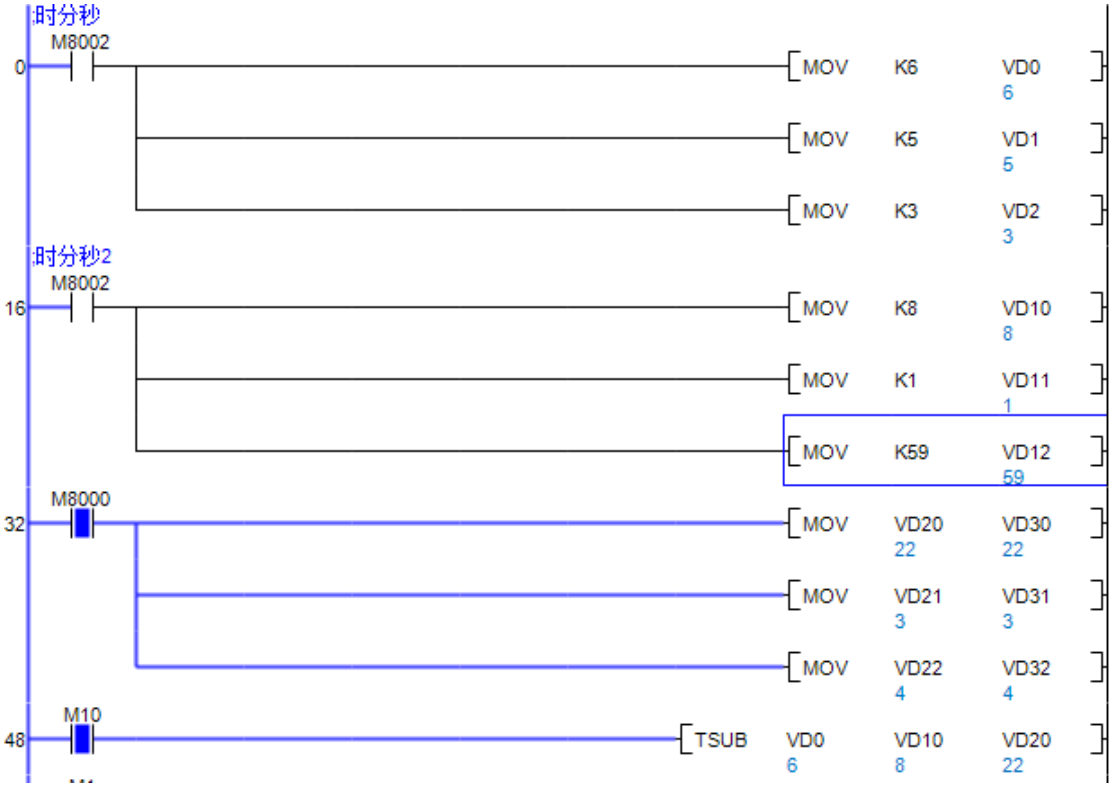


(S1,S1+1,S1+2)-(S2,S2+1,S2+2)
➜(D,D+1,D+2)

S1	(时)		S2	(时)		D	(时)
S1	+1(分)	-	S2	+1(分)	→	D	+1(分)
S1	+2(秒)		S2	+2(秒)		D	+2(秒)

“时”的范围为[0~23]
“分”的范围为[0~59]
“秒”的范围为[0~59]

3、程序举例



4、注意事项

相关标志位的动作：当运算结果小于 0 时，借位标志位变为 ON，从单纯的减法运算值中加上 24 个小时后，将该时间作为运算结果被保存。运算结果为 0(0 时 0 分 0 秒)时，零位标志位变为 ON。

3.13.5 HTOS 指令(时分秒数据的秒转换)

将[时、分、秒]单位的时间(时刻)数据转换成秒单位的数据的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
HTOS	HTOSP	DHTOS	DHTOSP	HTOS S D

操作数的数据类型如下表

操作数	内容	类型
S	保存转换前的时间(时刻)数据(时、分、秒)的软元件的起始编号	BIN16 位
D	保存转换后的时间(时刻)数据(秒)的软元件编号。	BIN16/32 位

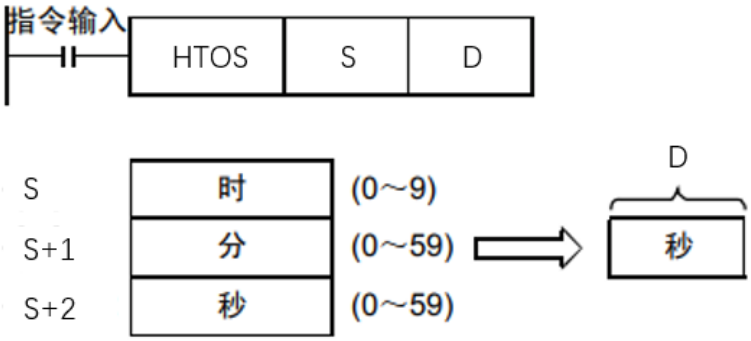
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明

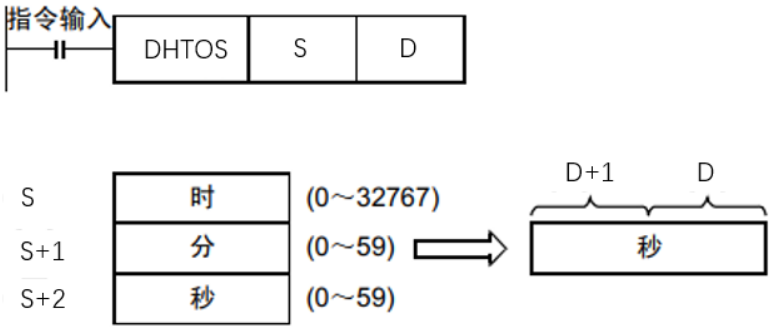
（1）16 位指令

将[S，S+1，S+2]的时间(时刻)数据(时、分、秒)换算成秒后，将结果保存到 D 中。

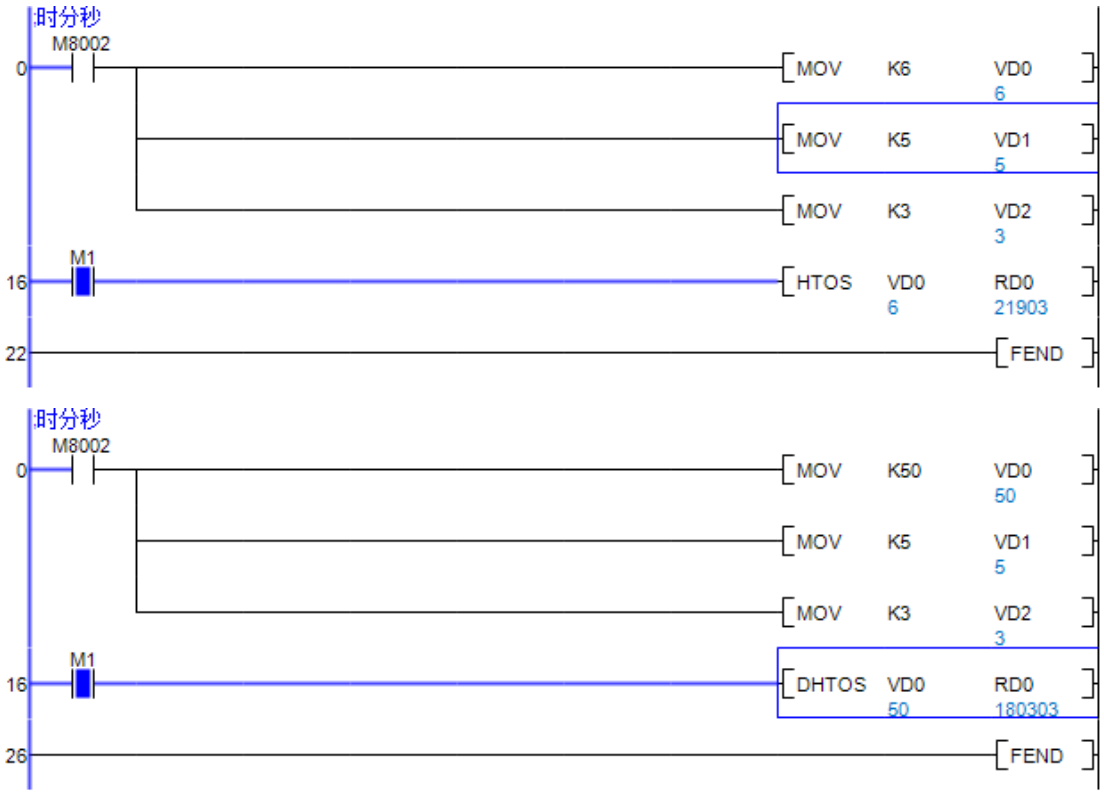


（2）32 位指令

将[S，S+1，S+2]的时间(时刻)数据(时、分、秒)换算成秒后，将结果保存到（D+1，D）中。



3、程序举例



4、注意事项

S、S+1、S+2 的数据超出与指令对应的可设定范围时，报错 6706。

3.13.6 STOH 指令(秒数据的时分秒转换)

将秒单位的时间(时刻)数据转换成[时、分、秒]单位的数据的指令。

1、指令格式

16bit 9 步		32bit		指令格式
STOH	STOHP	DSTOH	DSTOHP	STOH S D

操作数的数据类型如下表

操作数	内容	类型
S	保存转换前的时间(时刻)数据(秒)的软元件编号	BIN16/32 位
D	保存转换后的时间(时刻)数据(时、分、秒)的软元件起始编号	BIN16 位

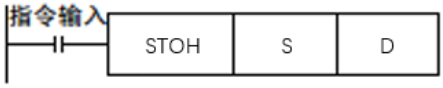
操作数的对象软元件如下表

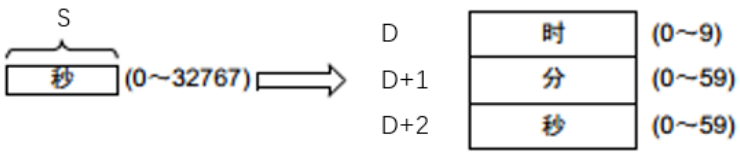
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明

（1）16 位指令

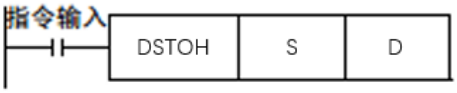
将 S 的秒数据换算成时、分、秒，其结果保存到[D，D+1，D+2](时、分、秒)中。

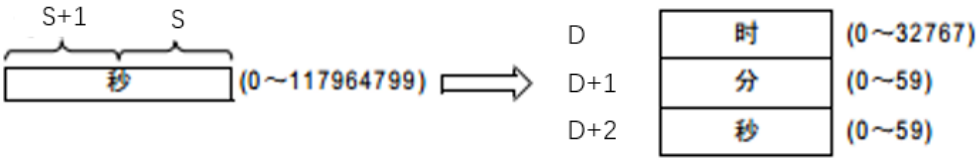




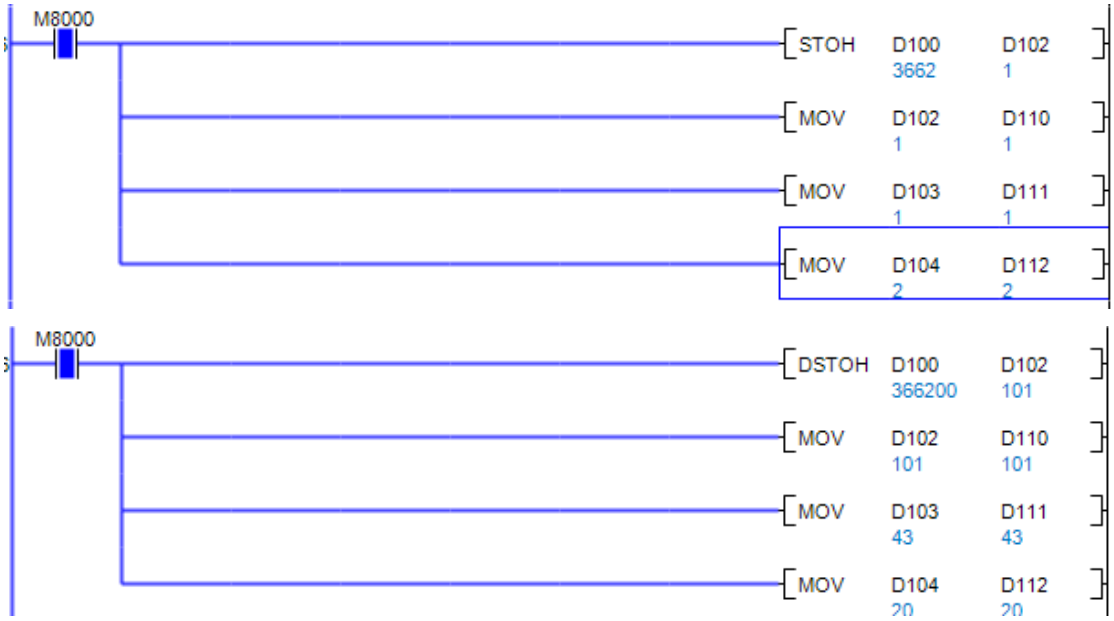
（2）32 位指令

将（S+1，S）的秒数据换算成时、分、秒，其结果保存到[D，D+1，D+2](时、分、秒)中。





3、程序举例



4、注意事项

S、S+1、S+2 的数据超出与指令对应的可设定范围时，报错 6706。

3.13.7 TRD 指令(读出时钟数据)

将可编程控制器内置实时时钟的时钟数据(D8013～D8019)按照下面的格式读出 D～D+6 中。

1、指令格式

16bit 3 步		32bit		指令格式
TRD	TRDP			TRD D

操作数的数据类型如下表

操作数	内容	类型
D	指定保存读出时间数据的起始软元件编号。(占用 7 点)	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P

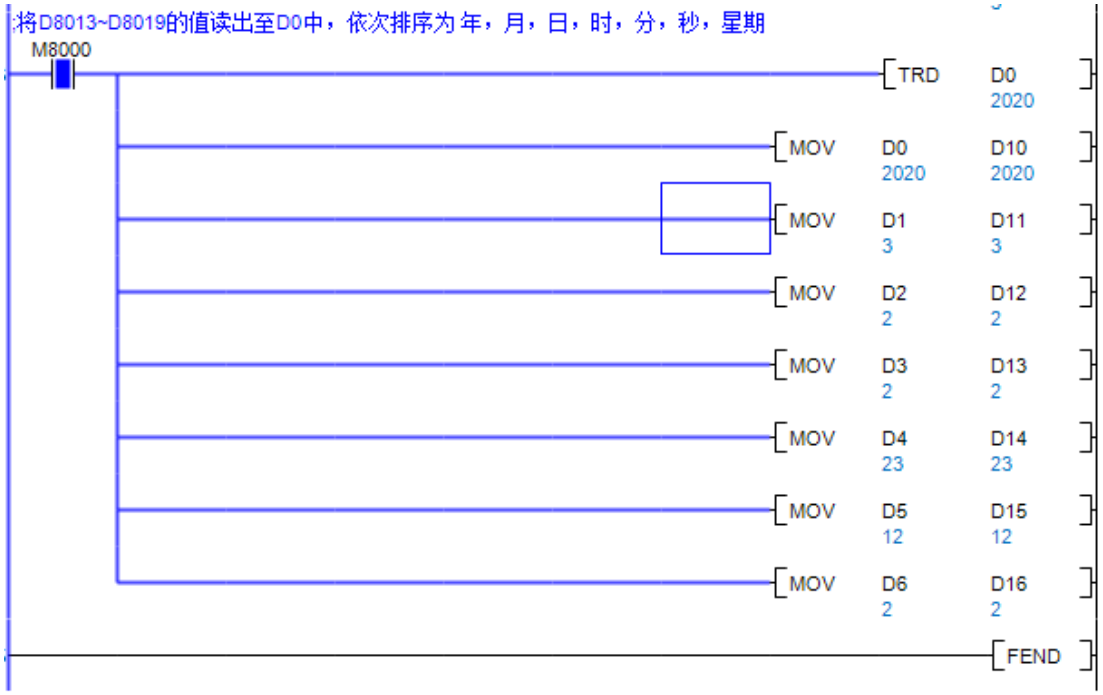
S					●	●	●	●	●	●			●	

2、功能和动作说明

16 位指令，将可编程控制器内置实时时钟的时钟数据(D8013～D8019)按照下面的格式读出到 D ～ D+6 中。

	软元件	项目	时钟数据		软元件	项目	
特殊数据寄存器	D8018	年（公历）	0~99（公历后 2 位数）	→	D0	年（公历）	读取用寄存器
	D8017	月	1~12	→	D1	月	
	D8016	日	1~31	→	D2	日	
	D8015	时	0~23	→	D3	时	
	D8014	分	0~59	→	D4	分	
	D8013	秒	0~59	→	D5	秒	
	D8019	星期	0（日）~6（六）	→	D6	星期	

3、程序举例



4、注意事项

D 占用 7 点软元件，请注意不要与机器其他控制中使用的软元件重复。

3.13.8 TWR 指令(写入时钟数据)

向可编程控制器内置实时时钟写入时钟数据的指令。

1、指令格式

16bit 3 步		32bit		指令格式
TWR	TWRP	\	\	TWR S

操作数的数据类型如下表

操作数	内容	类型
S	指定写入时间数据的源地址的起始软元件编号。(占用 7 点)	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	

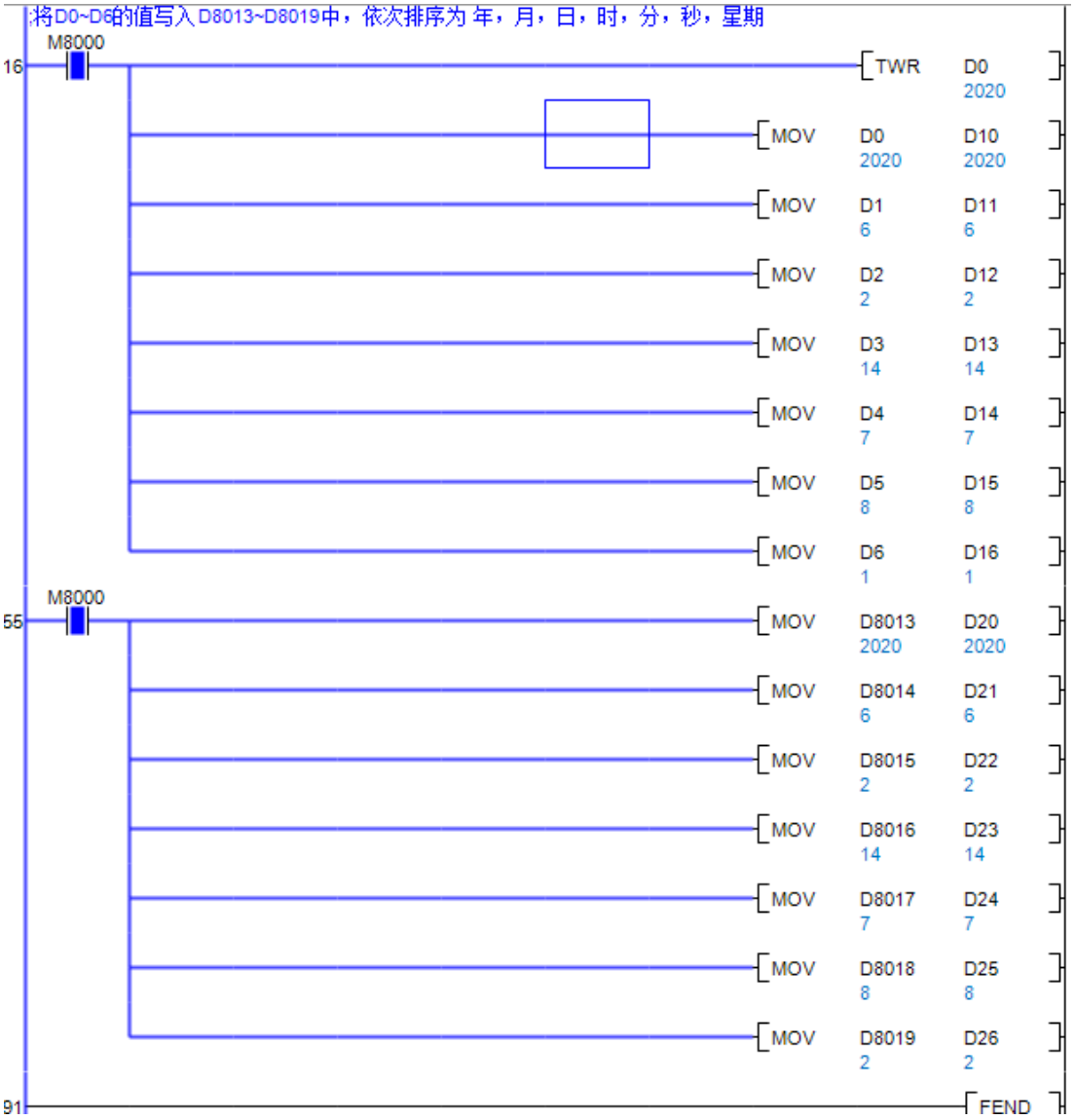
2、功能和动作说明

将设定的时钟数据 S ～ S+6 写入可编程控制器内置实时时钟的时钟数据(D8013～D8019)中。

设定时间	软元件	项目	时钟数据		软元件	项目	特殊数据
	D10	年（公历）	0~99（公历后 2 位数）	→	D8018	年（公历）	
	D11	月	1~12	→	D8017	月	
	D12	日	1~31	→	D8016	日	

用	D13	时	0~23	→	D8015	时	寄
的	D14	分	0~59	→	D8014	分	存
数	D15	秒	0~59	→	D8013	秒	器
据	D16	星期	0（日）~6（六）	→	D8019	星期	

3、程序举例



4、注意事项

（1）执行 TWR 指令后，实时时钟的时钟数据即刻被更改。因此，请先将快几分钟的时钟数据传送到 S～S+6 中，等到变成正确的时间时才执行指令。

（2）使用这个指令设定时钟数据(时间校准)时，不需要控制特殊辅助继电器 M8015(时间停止以及时间校准)。

（3）设定了不可能显示的日期时间数值时，不执行时钟数据的变更。此时，请设定正确的时钟数据后再次写入。

（4）星期(D8019)与写入数值无关,是根据日期内容自动补正。

（5）占用 S 开始的连续 7 点软元件，请注意不要与机器其他控制中使用的软元件重复。

（6）设定时间时，设定快 1 分钟的时间，等到达正确时间时使 TWR 指令前面的触发开关置位，以使适中数据更新。

3.13.9 HOUR 指令(计时表)

以 1 个小时为单位，对输入触点持续 ON 的时间进行累加检测的指令。

1、指令格式

16bit 7 步		32bit 12 步		指令格式
HOUR	HOURP	DHOUR	HOURP	HOUR S D1 D2

操作数的数据类型如下表

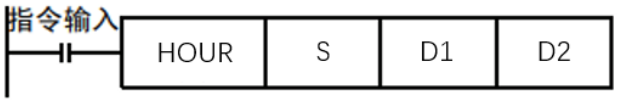
操作数	内容	类型
S	使 D2 为 ON 的时间(以 1 个小时为单位设定)	BIN16/32 位
D1	以 1 个小时为单位的当前值(指定停电保持用数据寄存器)	BIN16/32 位
D2	报警输出的起始编号	位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D1														
D2		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D1							●	●	●	●			●	
D2													●	

2、功能和动作说明

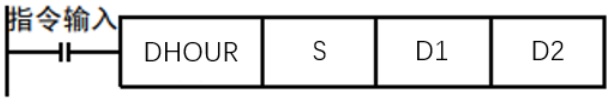
（1）16 位指令



当指令输入的累计 ON 时间超出了 S 时，D2 变为 ON。D1+1 中不满 1 个小时的当前值，以 1 秒单位被保存。

S：D2 变 ON 为止的时间以 1 个小时为单位指定。
D1：以 1 个小时为单位的当前值
D1+1 ：不满 1 个小时的当前值(1 秒单位)
D2：报警输出目标地址编号，当前值 D1 在 S 的指定时间以上时置 ON。

（2）32 位指令



（S+1，S）：D2 变为 ON 为止的时间指定，用 S1+1(高位)，S1(低位)指定。
（D1+1，D1）：以 1 个小时为单位的当前值保存在 D1+1(高位)，D1(低位)中。
D1+2 ：不满 1 个小时的当前值(1 秒单位)
D2：报警输出的指定，当前值 D1、D1+1 在 S 的指定时间以上时置 ON

3、程序举例

4、注意事项

（1）由于即使断开可编程控制器的电源后，也可以继续使用当前值数据，所以请在 D1 中指定停电保持用的数据寄存器。使用一般的数据寄存器时，由于可编程控制器的电源 OFF 和 STOP→RUN 的操作，当前值会被清除。

（2）报警输出 D2 为 ON 以后，测量仍能继续。再次开始进行测量的情况下，请清除 D1～D1+1(32 位指令为 D1～D1+2)的当前值。清除当前值时后，报警输出将变为 OFF。

（3）当前值 D1 达到最大值时停止测量。要继续测量时，请清除 D1～D1+1(32 位指令为 D1～D1+2)的当前值。

（4）D1 占用 2 个(16 位运算)或者 3 个(32 位运算)软元件。请注意不要与机器其他控制中使用的软元件重复。

3.13.10 TIM 指令(定时器)

局部定时器指令。

1、指令格式

16bit 7 步		32bit		指令格式
TIM	\	\	\	HOUR S S1 S2

操作数的数据类型如下表

操作数	内容	类型
S	定时器编号	BIN16
S1	定时设定长度	BIN16
D	定时基准(1 10 100 1000)	BIN16

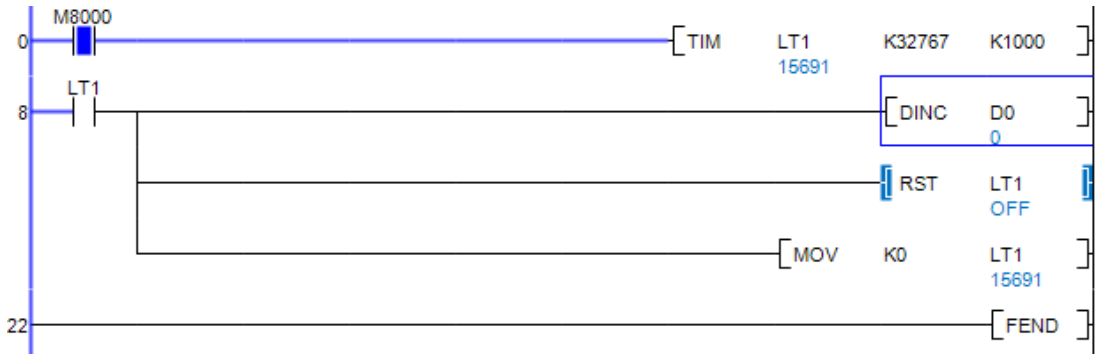
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
S1											●	●		
D											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S														
S1							●	●						
D														

2、功能和动作说明

当 LT1 以 S2 设定的时钟基准走时，到达设定 S1 的时，LT1ON。

程序举例



3.13.11 TIS 指令(周期定时器)

周期延指令

1、指令格式

16bit 7 步		32bit		指令格式
TIM	\	\	\	TIM S D

操作数的数据类型如下表

操作数	内容	类型
S	定时器编号	BIN16
D	扫描周期数量	BIN16

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S														
D							●	●						

2、功能和动作说明

根据 S2 设定的扫描周期数量，到达后 LT1 为 ON

3、程序举例

3.14 格雷码指令

3.14.1 GRY 指令(格雷码的转换)

将 BIN 值转换成格雷码后进行传送的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
GRY	GRYP	DGRY	DGRYP	GRY S D

操作数的数据类型如下表

操作数	内容	类型
S	转换源数据，或是保存转换源数据的字软元件	BIN16/32 位
D	保存转换后数据的字软元件	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	

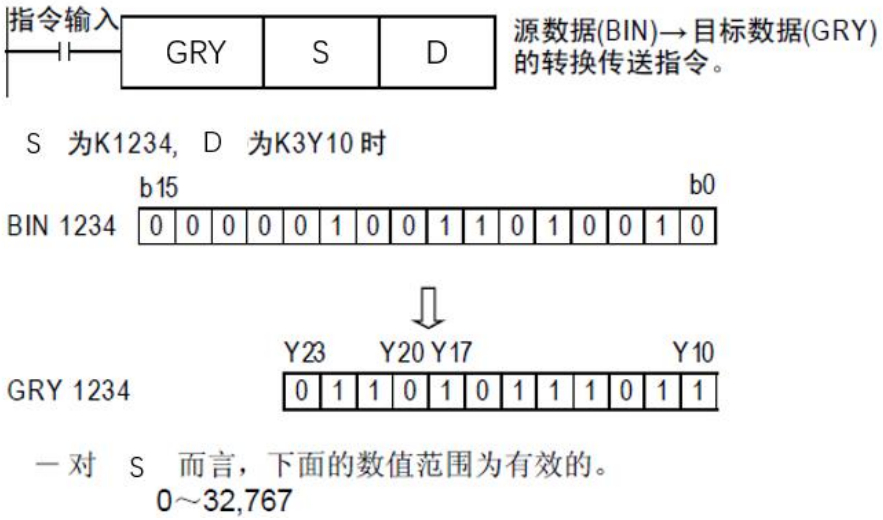
D		●	●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

源数据(BIN)→目标数据(GRY)的转换传送指令。格雷码的值只需要在原来的二进制的基础上右移一位再异或原来的二进制值即可得到。

3、程序举例

S 为 K1234，D 为 K3Y10 时



4、注意事项

（1）数据的转换速度取决于可编程控制器的扫描时间。

（2）16 位指令时，S 范围为 0~32767；32 位指令时，S 范围为 0~2147483647。

3.14.2 GBIN 指令(格雷码的逆转换)

将格雷码转换成 BIN 值后进行传送的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
GBIN	GBINP	DGBIN	DGBINP	GBIN S D

操作数的数据类型如下表

操作数	内容	类型
S	转换源数据，或是保存转换源数据的字软元件	BIN16/32 位
D	保存转换后数据的字软元件	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

源数据(GRY)→目标数据(BIN)的转换传送指令。二进制格雷码转换成自然二进制码,其法则是保留格雷码的最高位作为自然二进制码的最高位,而次高位自然二进制码为高位自然二进制码与次高位格雷码相异或,而自然二进制码的其余各位与次高位自然二进制码的求法相类似。

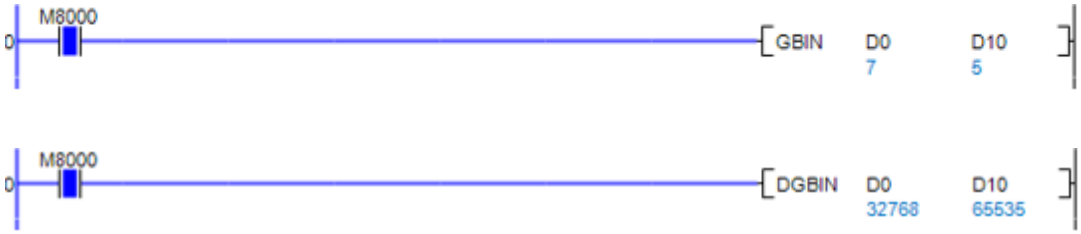
3、程序举例

S 为 K3X000, D 为 10 时

4、注意事项

16 位指令时，S 范围为 0~32767；32 位指令时，S 范围为 0~2147483647。

程序举例



3.15 其他指令

3.15.1 COMRD 指令(读出软元件的注释数据)

这个指令是将软元件的注释读取出来。

1、指令格式

16bit 5 步		32bit		指令格式
COMRD	COMRDP	\	\	COMRD S D

操作数的数据类型如下表

操作数	内容	类型
S	登录了要读出注释的软元件编号	BIN16

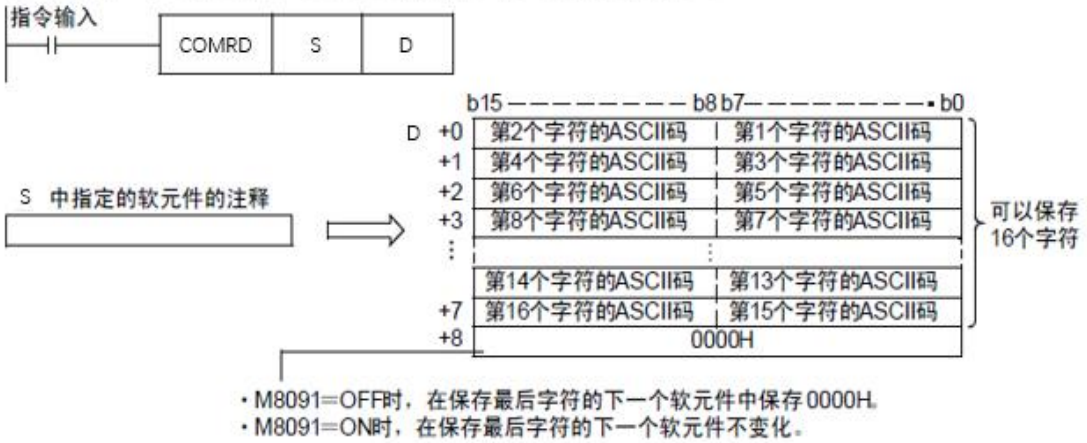
D	保存已经读出注释的软元件起始编号	BIN16

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
S	●	●	●			●	●	●	●					
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	

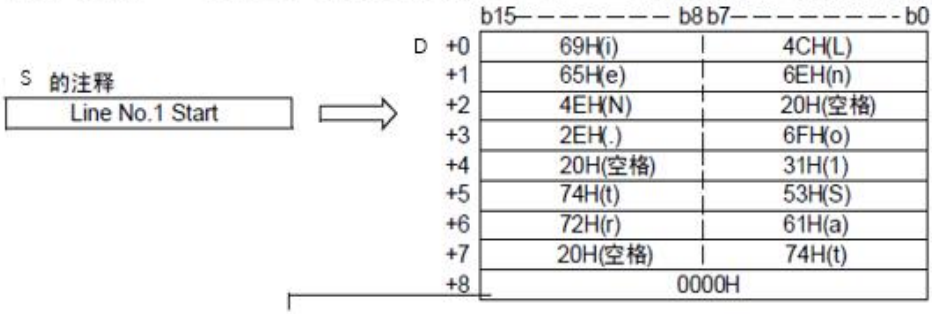
2、功能和动作说明

读出软元件 S 的注释后，以ASCII码保存在 D 开始的软元件中。



・M8091=OFF时，在保存最后字符的下一个软元件中保存0000H。
・M8091=ON时，在保存最后字符的下一个软元件不变化。

例如，软元件 S 的注释为“Line NO.1 Start”时，从 D 开始的软元件中保存如下所示的内容。



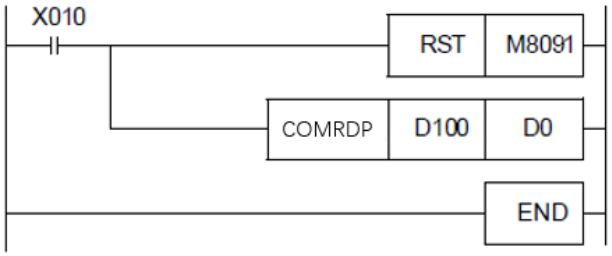
・M8091=OFF时，在保存最后字符的下一个软元件中保存0000H。
・M8091=ON时，在保存最后字符的下一个软元件不变化。

ON/OFF 状态	处理内容

M8091=OFF	在保存最后注释字符的软元件的下一个软元件中保存 0000H。
M8091=ON	保存最后注释字符的软元件的下一个软元件不变化。

3、程序举例

X010为ON后，D100中设定的注释“Target Line A”会以ASCII码保存到从D0开始的软元件中的程序。(M8091＝OFF时)



D100的注释

Target Line A

	b15 b8	b7 b0
D0	61H(a)	54H(T)
D1	67H(g)	72H(r)
D2	74H(t)	65H(e)
D3	4CH(L)	20H(空格)
D4	6EH(n)	69H(i)
D5	20H(空格)	65H(e)
D6	20H(空格)	41H(A)
D7	20H(空格)	20H(空格)
D8	0000H	

4、注意事项

（1）请在S的软元件中，指定在可编程控制器中登录(写入)了注释的软元件编号。
（2）在软元件中未登录(写入)注释时，注释的字符数(半角16个字符)全部都以“20H”(空格)保存到D开始的软元件中。

3.15.2 RND 指令(产生随机数)

这个指令产生0～32767的伪随机数，将其数值作为随机数保存到D中。
在伪随机数系列中，每次计算出随机数的原始值，然后使用这随机数的原始值计算出伪随机数。

1、指令格式

16bit 3 步		32bit		指令格式
RND	RNDP	\	\	RND D

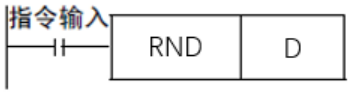
操作数的数据类型如下表

操作数	内容	类型
D	保存已经读出注释的软元件起始编号	BIN16

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明



3、程序举例

3.15.3 DUTY 指令(产生定时脉冲)

1、 指令格式

16bit 7 步		32bit		指令格式
DUTY	\	\	\	DUTY N1 N2 D

操作数的数据类型如下表

操作数	内容	类型
n1	ON 的扫描次数(运算周期) [n1>0]	BIN16
n2	OFF 的扫描次数(运算周期)[n2>0]	BIN16
D	定时时钟输出的目标地	位

操作数的对象软元件如下表

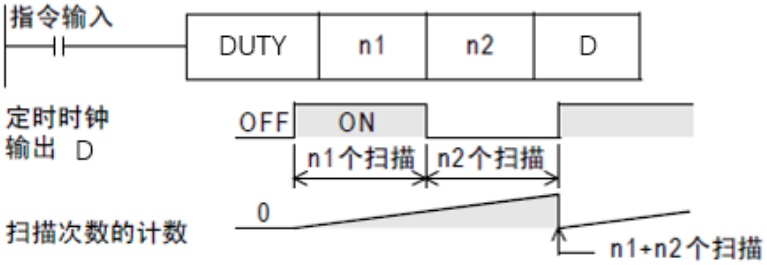
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”

n1											●	●		
n2											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
n1					●	●	●	●	●	●	●	●		
n2					●	●	●	●	●	●	●	●		
D													●	

2、功能和动作说明

16位运算(DUTY)

1) 定时时钟输出 D 是按照n1个扫描ON，n2个扫描OFF的方式进行ON/OFF。



2) 请在定时时钟的输出的目标地址中，指定M8330～M8334。

3) 与定时时钟输出的目标地址对应的扫描数的计数值被保存到D8330～D8334中。

扫描数的计数值 D8330～D8334，在计数值变为 n1＋n2 时，或者在指令输入(指令)变为 ON 时被复位。

定时时钟输出的目标地址	对扫描数计数用的软元件
M8330	D8330
M8331	D8331
M8332	D8332
M8333	D8333
M8334	D8334

4) 在指令输入的上升沿开始动作，在END指令处， ON/OFF定时时钟输出。

此外，指令输入即使被切断动作也不停止。STOP 时，通过中断，或是断电时停止。

5) 将n1、n2设定为0时，如下表所示。

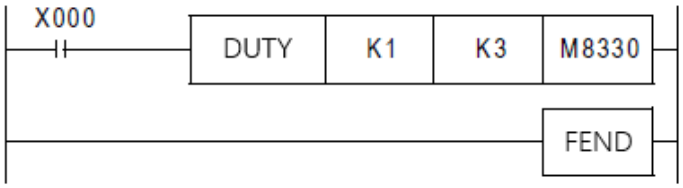
n1、n2的状态	ON/OFF的状态
n1＝0，n2≧0	固定为OFF
n1＝0，n2≧0	固定为 ON

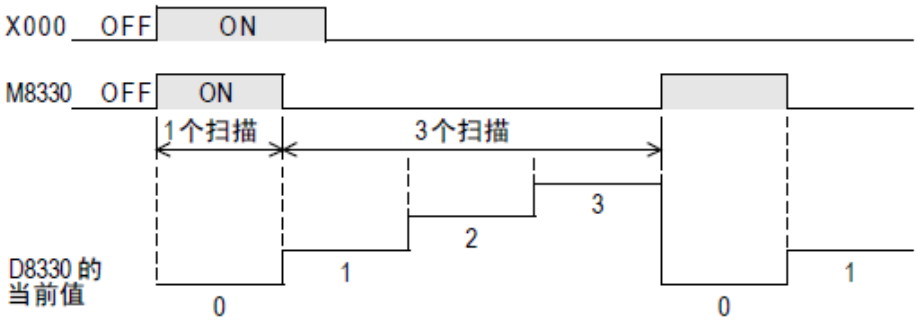
6)相关软元件

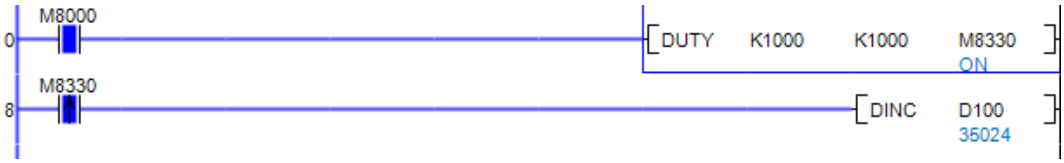
软元件	名称	内容
M8330	定时时钟输出 1	DUTY 指令的定时时钟输出
M8331	定时时钟输出 2	
M8332	定时时钟输出 3	
M8333	定时时钟输出 4	
M8334	定时时钟输出 5	
D8330	定时时钟输出 1 用扫描次数的计数	DUTY 指令的定时时钟输出 1 用的扫描次数的计数值
D8331	定时时钟输出 2 用扫描次数的计数	DUTY 指令的定时时钟输出 2 用的扫描次数的计数值
D8332	定时时钟输出 3 用扫描次数的计数	DUTY 指令的定时时钟输出 3 用的扫描次数的计数值
D8333	定时时钟输出 4 用扫描次数的计数	DUTY 指令的定时时钟输出 4 用的扫描次数的计数值
D8334	定时时钟输出 5 用扫描次数的计数	DUTY 指令的定时时钟输出 5 用的扫描次数的计数值

3、程序举例

X000为ON后，M8330 1个扫描为ON，3个扫描为OFF的程序。







4、注意事项

（1）该指令能使用5次(点)。
（2）但是，在多个 DUTY 指令中不能使用相同的定时时钟输出目标地址 D。

3.15.4 CRC 指令(CRC 运算)

16bit 7 步		32bit		指令格式
CRC	CRCP	\	\	CRC S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存作为 CRC 值生成对象的数据的软元件起始编号	BIN16
D	保存被生成的 CRC 值的软元件编号	BIN16
n	要计算CRC值的8位数据(字节)数，或是保存数据数的软元件编号	BIN16

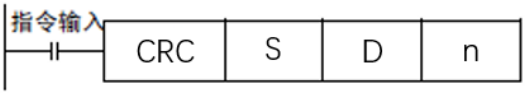
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D		●	●	●	●	●	●	●	●	●	●	●	●	
n							●	●	●	●			●	

2、功能和动作说明

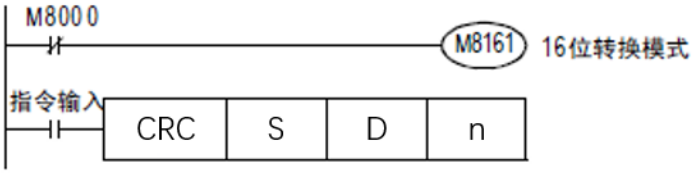
16 位运算

以S中指定的软元件为起始的n点8位数据(字节单位)，对其生成CRC值后保存到D中。
在这个指令中有8位和16位的转换模式，根据M8161的ON/OFF来切换转换模式。
此外，为了生成 CRC 值(CRC-16)，使用了[X16+X15+X2+1]的生成多项式。



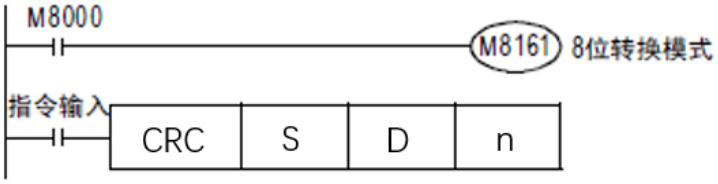
16 位转换模式[M8161＝OFF]

在16位模式下，对S软元件的高8位(字节)和低8位(字节)进行运算。
在 D 指定的 1 点软元件的 16 位中保存运算结果。



			例如 S ＝ D100 D ＝ D0 n ＝ 6		
			软元件	对象数据内容	
				8 位	16 位
保存生成 CRC 值的对象数据的地址	S	低字节	D100 低字节	01H	0301H
		高字节	D100 高字节	03H	
	S+1	低字节	D101 低字节	03H	0203H
		高字节	D101 高字节	02H	
	S+2	低字节	D102 低字节	00H	1300H
		高字节	D102 高字节	14H	
	≀	≀	~		
	S+n/2-1	低字节	-		
		高字节			
保存 CRC 值的地址	D	低字节	D0 低字节	E4H	4114H
		高字节	D0 高字节	41H	

8位转换模式[M8161＝ON]
在8位转换模式下，仅对S软元件的低8位(字节)执行运算。
计算结果使用D指定的软元件开始的2 点，在D中保存低8 位(字节)，在D+1 中保存高8位(字节)。

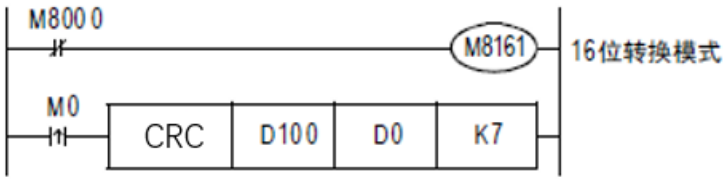


			例如 S = D100 D = D0 n = 6	
			软元件	对象数据内容
保存生成 CRC 值的对象数据的地址	S	低字节	D100 低字节	01H
	S+1	低字节	D101 低字节	03H
	S+2	低字节	D102 低字节	03H
	S+3	低字节	D103 低字节	02H
	S+4	低字节	D104 低字节	00H
	S+5	低字节	D105 低字节	14H
	≀		~	
	S+n-1	低字节	~	
保存 CRC 值的地址	D	低字节	D0 低字节	E4H
	D+1	低字节	D1 低字节	41H

2、 程序举例

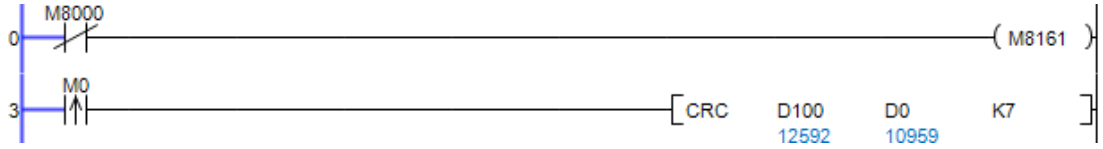
16 位模式下

M0 为 ON 时，生成 D100～D106 中保存的 ASCII 码[0123456]的 CRC 值后，保存到 D0 中的程序。

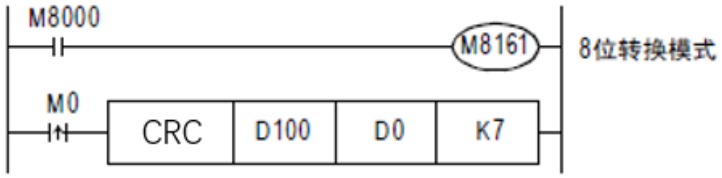


	数据的内容			
			对象数据	
保存生成 CRC 值的对象数据的地址	D100	3130H	低字节	30H
			高字节	31H
	D101	3332H	低字节	32H
			高字节	33H

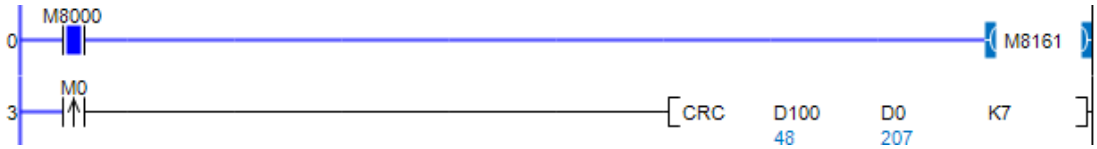
	D102	3534H	低字节	34H
			高字节	35H
	D103	3736H	低字节	36H
			~	~
保存 CRC 值的地址	D0	2ACFH	低字节	CFH
			高字节	2AH



8 位模式下



	数据的内容		
保存生成 CRC 值的对象数据的地址	D100	低字节	30H
	D101	低字节	31H
	D102	低字节	32H
	D103	低字节	33H
	D104	低字节	34H
	D105	低字节	35H
	D106	低字节	36H
保存 CRC 值的地址	D0	低字节	CFH
	D1	低字节	2AH



D0 为 207 D1 为 42

4、注意事项

（1）在这个指令中，使用CRC值(CRC-16)的生成多项式[X16+X15+X2+1]，此外，针对在CRC

值，还有各种标准化的生成多项式。请注意，如果使用了不同的生成多项式，会产生完全不同的CRC值。

（2）以下一些情况下会发生运算错误，错误标志位M8067置ON，错误代码保存在D8067中。

- S、D中使用的位软元件的位数指定，指定了4位数以外的值时。(错误代码：K6706)
- n在指定范围(1～256)以外时。(错误代码：K6706)
- S+n-1、D+1 超出软元件范围时。(错误代码：K6706)

3.16 数据块指令

3.16.1 BK+指令(数据块的加法运算)

数据块的 BIN 加法运算的指令。

1、指令格式

16bit 9 步		32bit 17 步		指令格式
BK+	BK+P	DBK+	DBK+P	BK+ S D

操作数的数据类型如下表

操作数	内容	类型
S1	保存执行加法运算的数据的软元件起始编号	BIN16/32 位
S2	执行加法运算的常数，或是保存执行加法运算的数据的软元件起始编号	BIN16/32 位
D	保存运算结果的软元件起始编号	BIN16/32 位
n	数据的个数	BIN16/32 位

操作数的对象软元件如下表

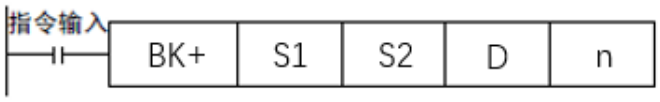
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
S2											●	●		
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1					●	●	●	●	●	●			●	

S2					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●			●	

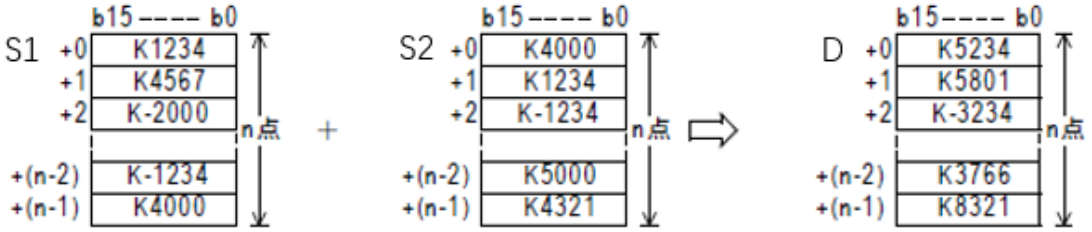
2、功能和动作说明

（1）16 位指令

将 S1 开始 n 点 16 位数据和 S2 开始的 n 点 16 位数据(BIN)进行加法运算后，将运算结果保存到 D 开始的 n 点中。



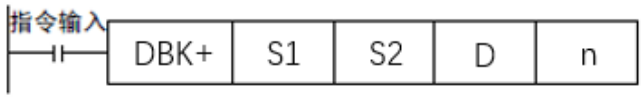
1） 将S1开始n点16位数据和S2开始的n点16位数据(BIN)进行加法运算后，将运算结果保存到D开始的 n 点中。



可以在 S2 中直接指定-32767~32767(16 位)的常数。

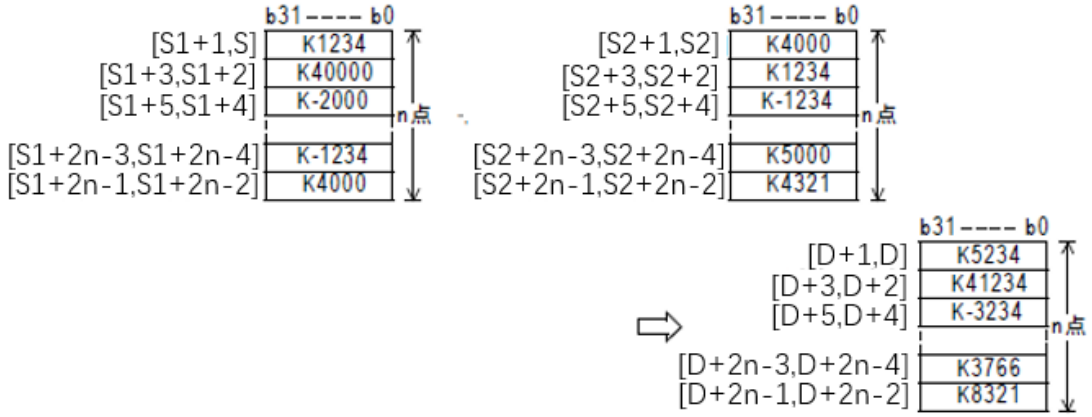
（2）32 位指令

将（S1+1，S1）开始 2n 点 32 位数据和（S2+1，S2）开始的 2n 点 32 位数据(BIN)进行加法运算后，将运算结果保存到（D+1，D）开始的 2 n 点中。

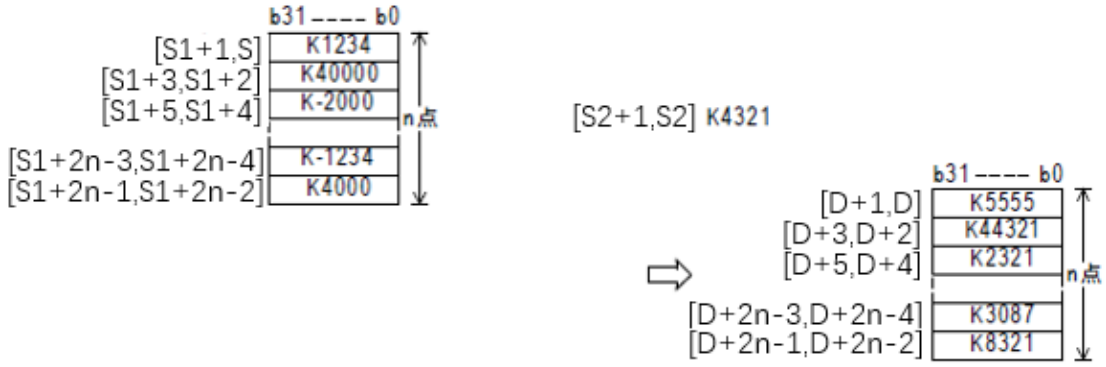


1） 将[S1+1,S1]开始的2n点32位数据和[S2+1，S2]开始的2n点32位数据(BIN)进行加法运算

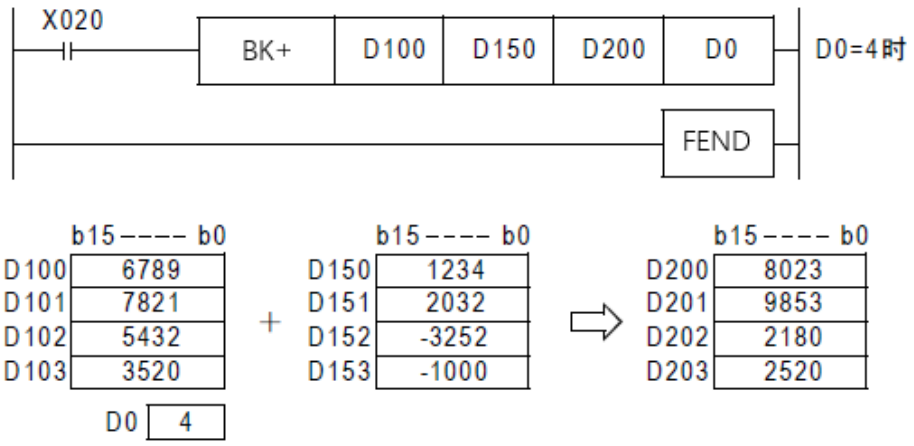
后，将运算结果保存到[D +1，D]开始的2n点中。

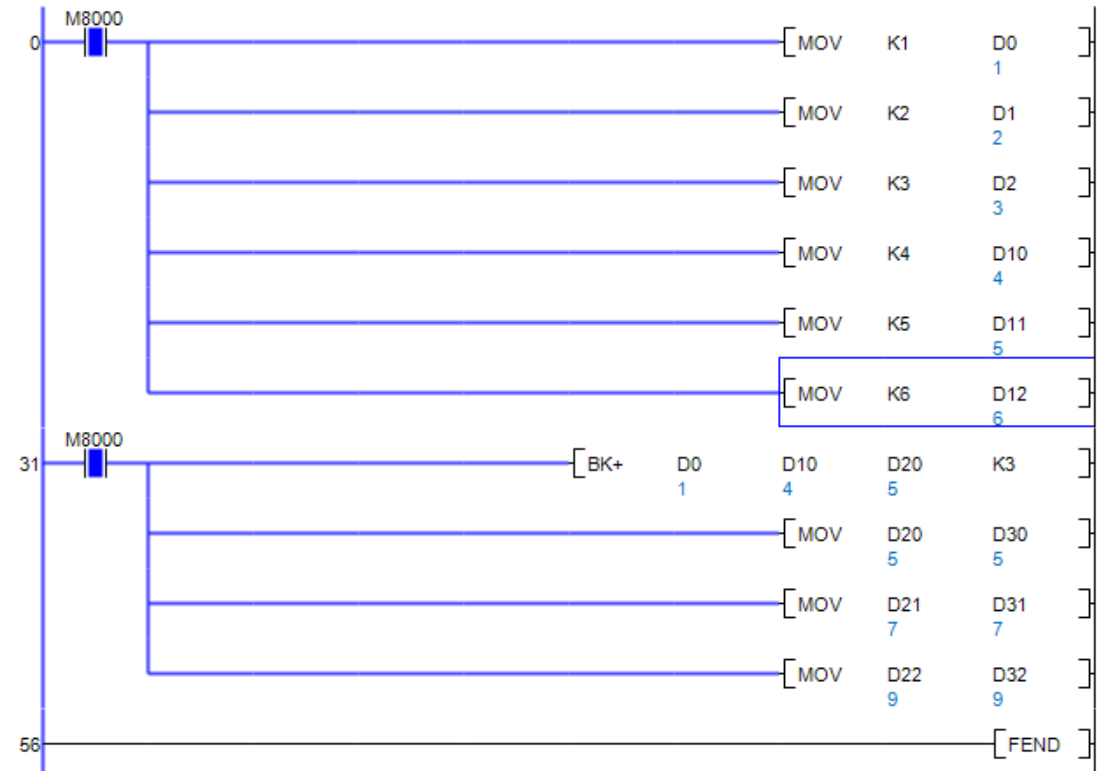


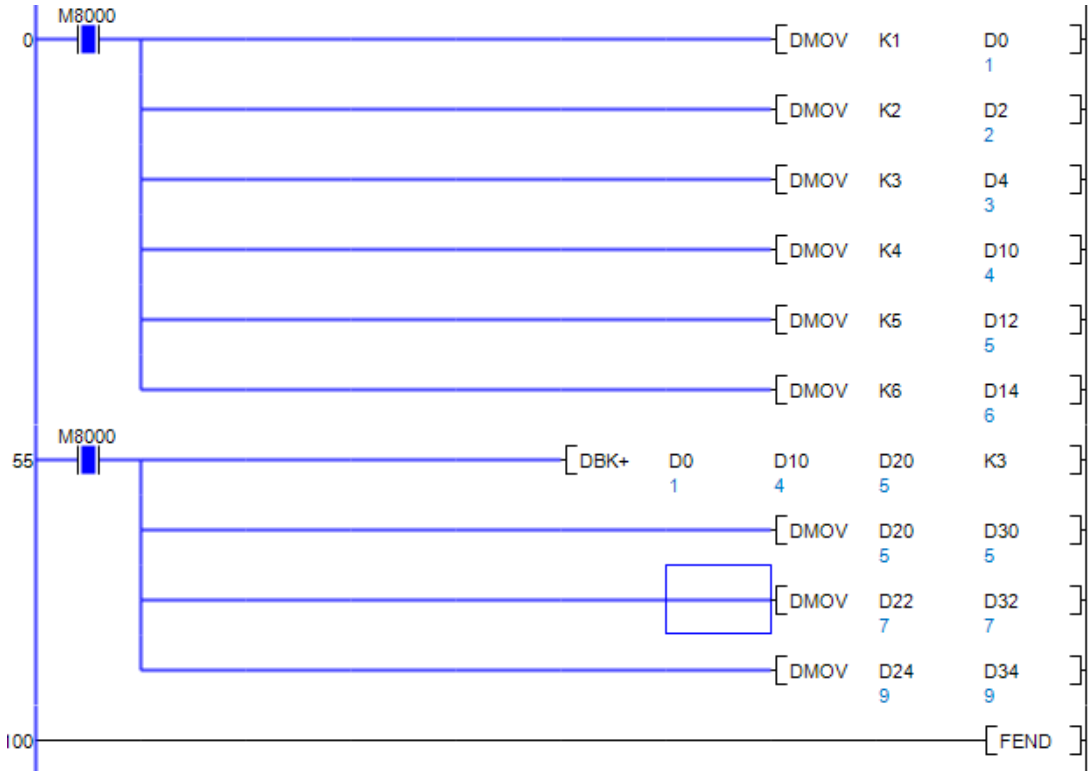
2) 可以在[S2+1, S2]中直接指定-2, 147, 483, 648～2, 147, 483, 647(32位)的常数。



3、 程序举例




M8000
0
MOV K1 D0
1
MOV K2 D1
2
MOV K3 D2
3
MOV K4 D10
4
MOV K5 D11
5
MOV K6 D12
6
M8000
31
BK+ D0 D10 D20 K3
1 4 5
MOV D20 D30
5 5
MOV D21 D31
7 7
MOV D22 D32
9 9
56
FEND


M8000
0
DMOV K1 D0
1
DMOV K2 D2
2
DMOV K3 D4
3
DMOV K4 D10
4
DMOV K5 D12
5
DMOV K6 D14
6
M8000
55
DBK+ D0 D10 D20 K3
1 4 5
DMOV D20 D30
5 5
DMOV D22 D32
7 7
DMOV D24 D34
9 9
100
FEND

4、注意事项

（1）运算结果中产生数据上溢出、下溢出时，进位标志不置位。

— 16位运算时

K32767(H7FFF) + K2(H0002) → K-32767(H8001)

K-32768(H8000) + K-2(HFFFE) → K32766(H7FFE)

— 32位运算时

K2,147,483,647(H7FFFFFFF) + K2(H00000002) → K-2,147,483,647(H80000001)

K-2,147,483,648(H80000000) + K-2(HFFFFFFFE) → K2,147,483,646(H7FFFFFFE)

（2）出现以下情况会报错 6706。

•S1、S2、D 开始的 n 点(32 位运算时为 2n 点)软元件超出相应的软元件范围时。

•S1 开始的 n 点软元件和 D 开始的 n 点软元件重复时(32 位运算时为 2n 点)。(错误代码: K6706)

•S2 开始的 n 点软元件和 D 开始的 n 点软元件重复时(32 位运算时为 2n 点)。(错误代码: K6706)

3.16.2 BK-指令(数据块的减法运算)

数据块的 BIN 减法运算的指令。

1、指令格式

16bit 9 步		32bit 17 步		指令格式
BK-	BK-P	DBK-	DBK-P	BK- S D

操作数的数据类型如下表

操作数	内容	类型
S1	保存执行减法运算的数据的软元件起始编号	BIN16/32 位
S2	执行减法运算的常数，或是保存执行减法运算的数据的软元件起始编号	BIN16/32 位
D	保存运算结果的软元件起始编号	BIN16/32 位
n	数据的个数	BIN16/32 位

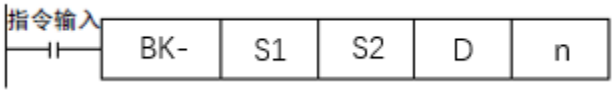
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
S2											●	●		

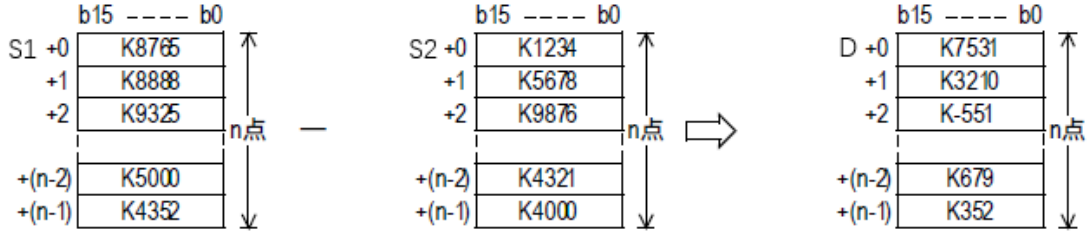
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1					●	●	●	●	●	●			●	
S2					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●			●	

2、功能和动作说明

（1）16 位指令

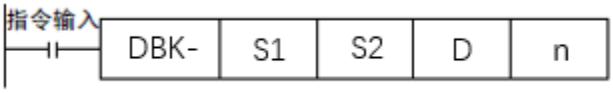


1)将 S1 开始 n 点 16 位数据和 S2 开始的 n 点 16 位数据(BIN)进行减法运算后，将运算结果保存到 D 开始的 n 点中。

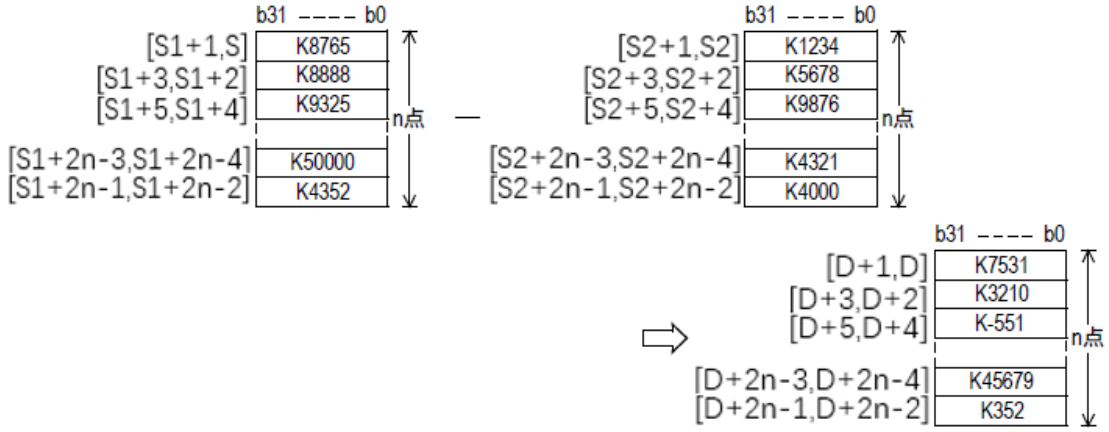


2) 可以在 S2 中直接指定-32768～32767(16 位)的常数。

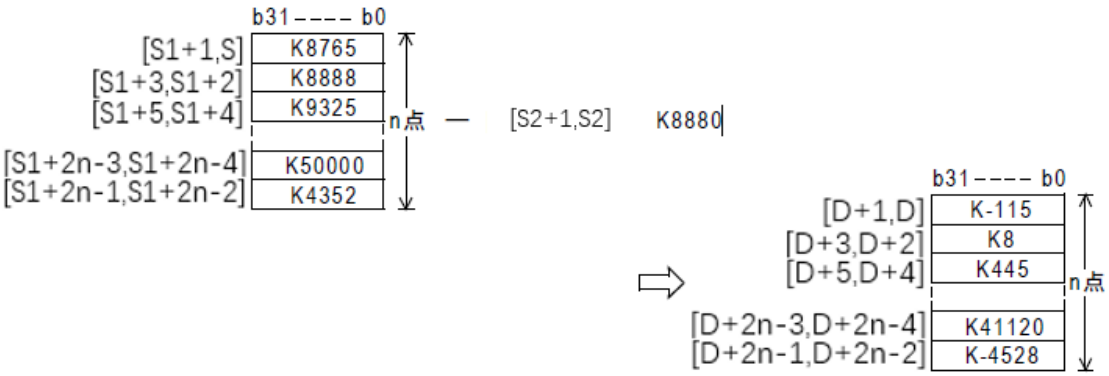
（2）32 位指令



1)将（S1+1，S1）开始 2n 点 32 位数据和（S2+1，S2）开始的 2n 点 32 位数据(BIN)进行减法运算后，将运算结果保存到（D+1，D）开始的 2 n 点中。

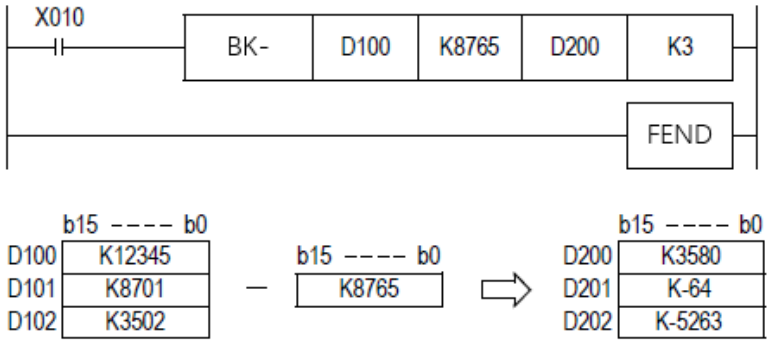


2) 可以在[S2+1,S2]中直接指定-2,147,483,648～2,147,483,647(32 位)的常数。

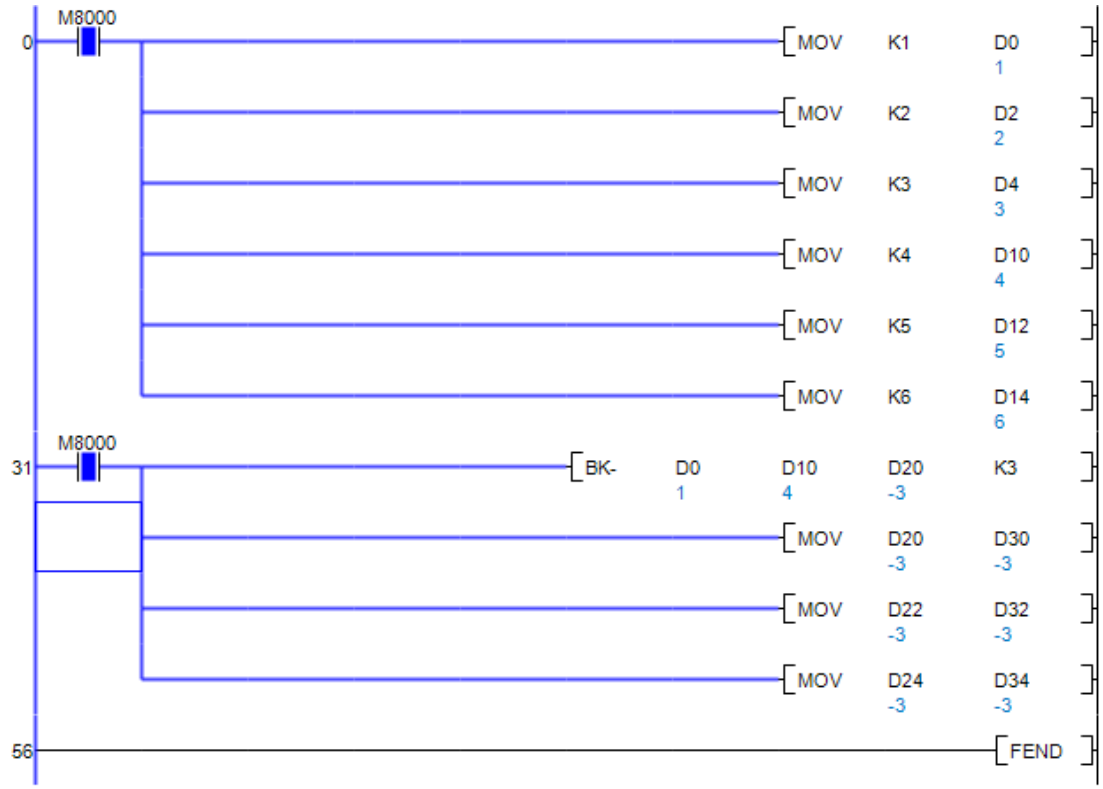


3、程序举例

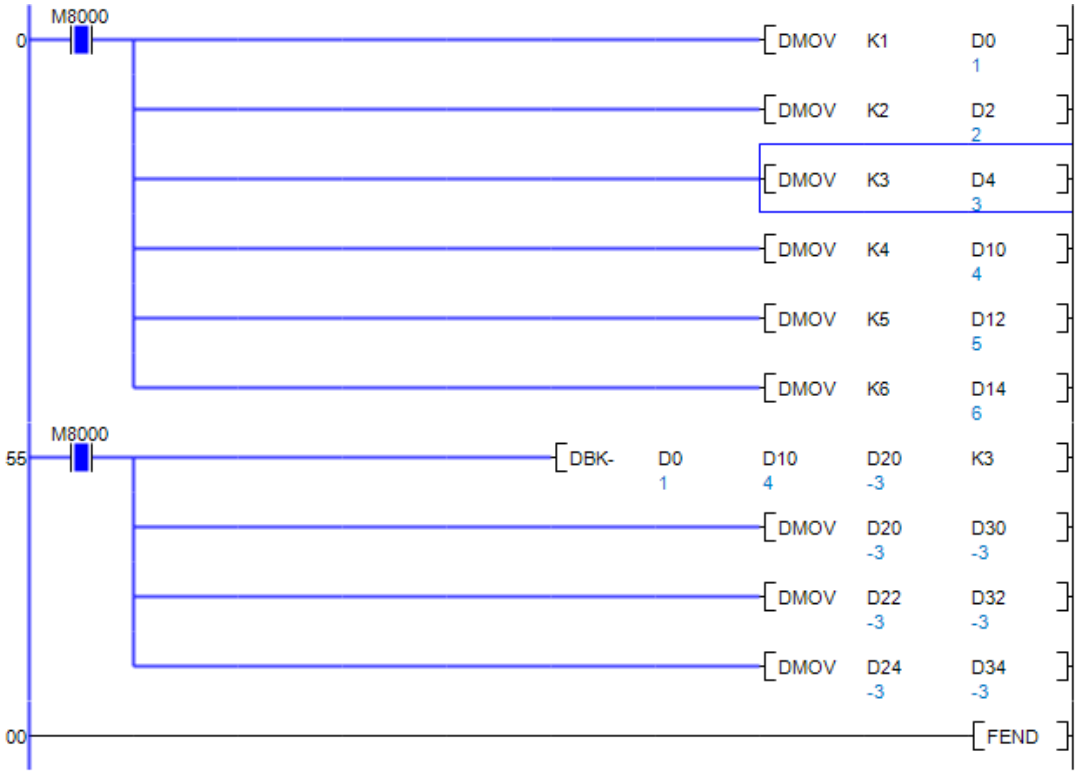
当 X010 为 ON 时，从 D100 开始的 3 点数据和常数 8765 相减后，将其结果保存到 D200 以后的软元件中的程序。



当上电后 D0-D4,D1-D5,D2-D6



当上电后 D0-D10,D2-D12,D4-D14



4、注意事项

（1）运算结果中产生数据上溢出、下溢出时，进位标志不置位。

— 16 位运算时

K-32768(H8000) - K2(H0002) → K32766(H7FFE)
K32767(H7FFF) - K-2(HFFFE) → K-32767(H8001)
— 32位运算时
K-2,147,483,648(H80000000) - K2(H00000002) → K2,147,483,646(H7FFFFFFE)
K2,147,483,647(H7FFFFFFF) - K-2(HFFFFFFFE) → K-2,147,483,647(H80000001)

（2）出现以下情况会报错 6706。

•S1、S2、D 开始的 n 点(32 位运算时为 2n 点)软元件超出相应的软元件范围时。
•S1 开始的 n 点软元件和 D 开始的 n 点软元件重复时(32 位运算时为 2n 点)。(错误代码: K6706)
•S2 开始的 n 点软元件和 D 开始的 n 点软元件重复时(32 位运算时为 2n 点)。(错误代码: K6706)

3.16.3 BKCMP 系列指令(数据块的比较)

这些指令按照各个指令的比较条件来比较数据块。

1、指令格式

16bit 9 步		32bit 17 步		指令格式
BKCMP=	BKCMP=P	DBKCMP=	DBKCMP=P	BKCMP= S1 S2 D n

16bit 9 步		32bit 17 步		指令格式
BKCMP>	BKCMP>P	DBKCMP>	DBKCMP>P	BKCMP> S1 S2 D n

16bit 9 步		32bit 17 步		指令格式
BKCMP<	BKCMP<P	DBKCMP<	DBKCMP<P	BKCMP< S1 S2 D n

16bit 9 步		32bit 17 步		指令格式
BKCMP<>	BKCMP<>P	DBKCMP<>	DBKCMP<>P	BKCMP<> S1 S2 D n

16bit 9 步		32bit 17 步		指令格式
BKCMP<=	BKCMP<=P	DBKCMP<=	DBKCMP<=P	BKCMP<= S1 S2 D n

16bit 9 步		32bit 17 步		指令格式
BKCMP>=	BKCMP>=P	DBKCMP>=	DBKCMP>=P	BKCMP>= S1 S2 D n

操作数的数据类型如下表

操作数	内容	类型
S1	比较值或是保存比较值的软元件编号	BIN16/32 位
S2	保存比较源数据的软元件起始编号	BIN16/32 位
D	保存比较结果的软元件起始编号	BIN16/32 位
n	要比较的数据数	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		

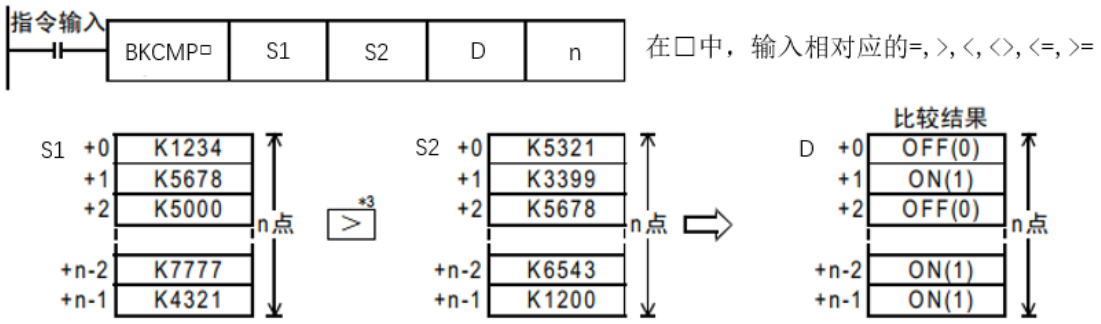
S2														
D		●	●			●	●	●	●	●				
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1					●	●	●	●	●	●			●	
S2					●	●	●	●	●	●			●	
D													●	
n							●	●	●	●			●	

2、功能和动作说明

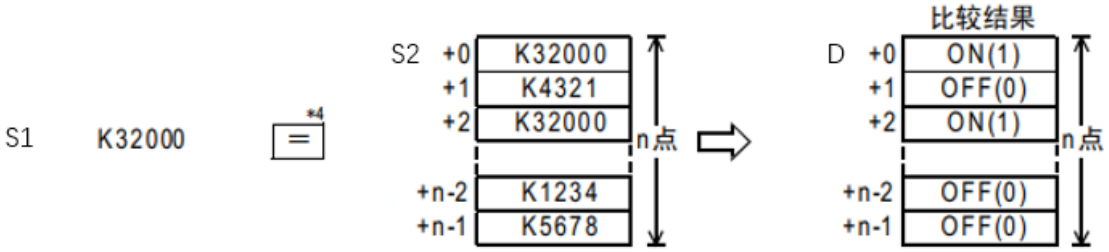
（1）16 位指令

将 S1 开始 n 点 16 位数据和 S2 开始的 n 点 16 位数据(BIN)进行比较后，将比较结果保存到 D 开始的 n 点中。

以 16 位指令 BKCMP>指令为例。

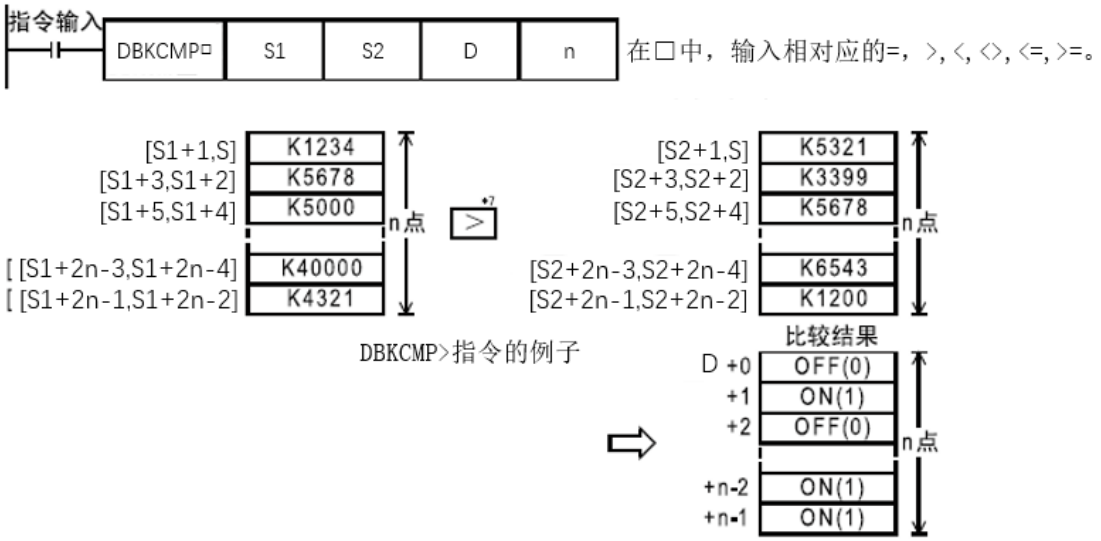


可以在 S1 中直接指定常数，以 BKCMP=指令为例。

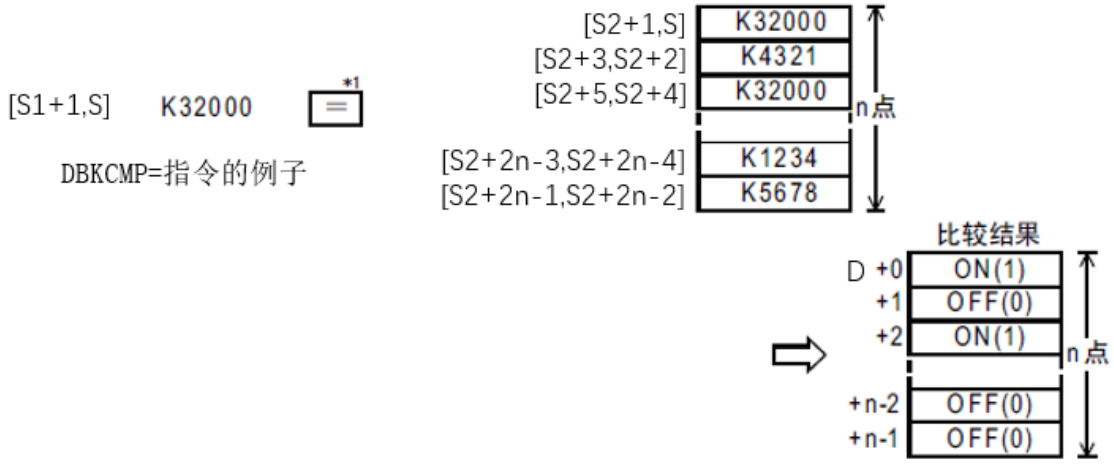


（2）32 位指令

将（S1+1，S1）开始 2n 点 32 位数据和（S2+1，S2）开始的 2n 点 32 位数据(BIN)进行比较后，将比较结果保存到（D+1，D）开始的 2 n 点中。



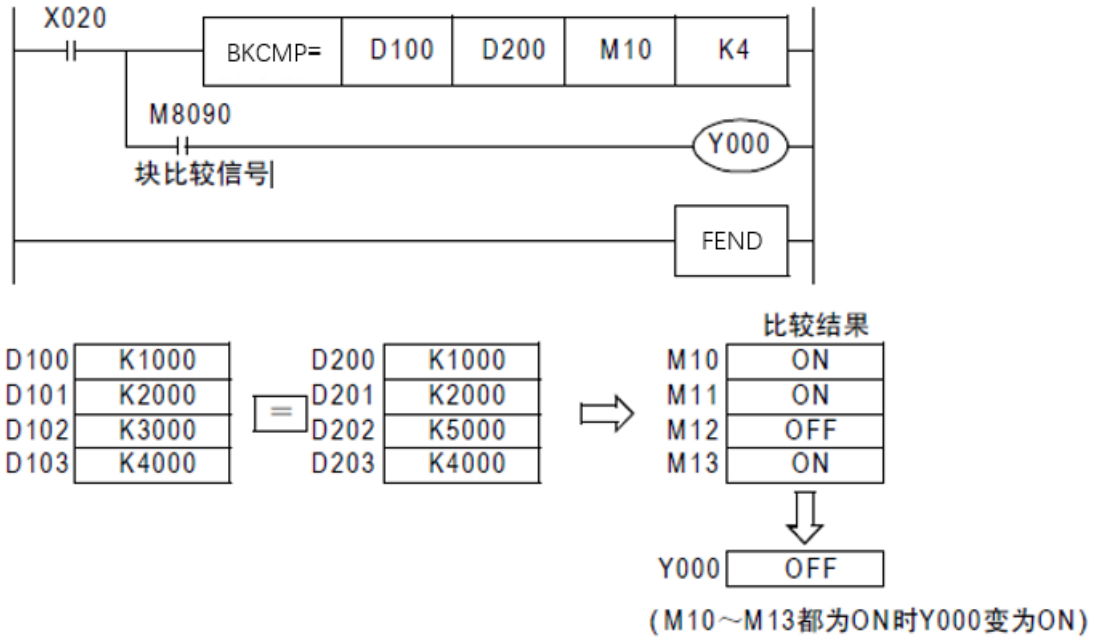
可以在[S1 +1，S1]中直接指定常数。



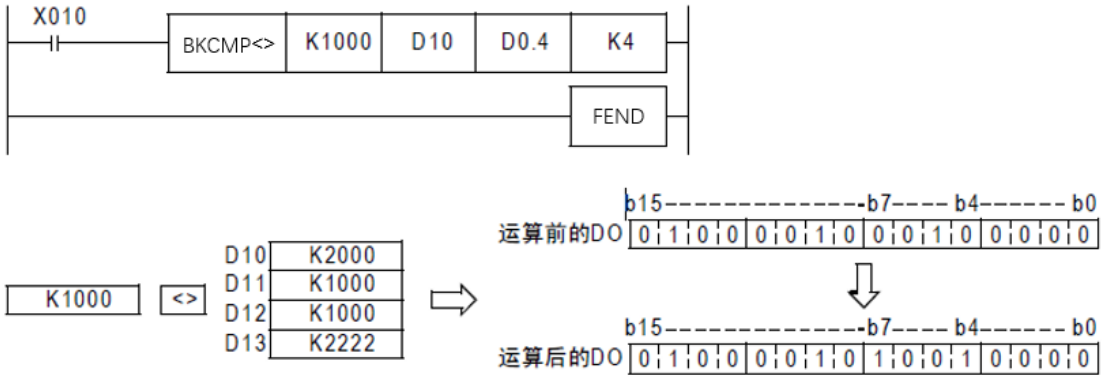
3、程序举例

当X020为ON时，使用BKCMP=指令对D100开始的4点16位数据(BIN)和D200开始的4点16位数据(BIN)进行比较，并将其结果保存到M10开始的4点软元件中的程序。

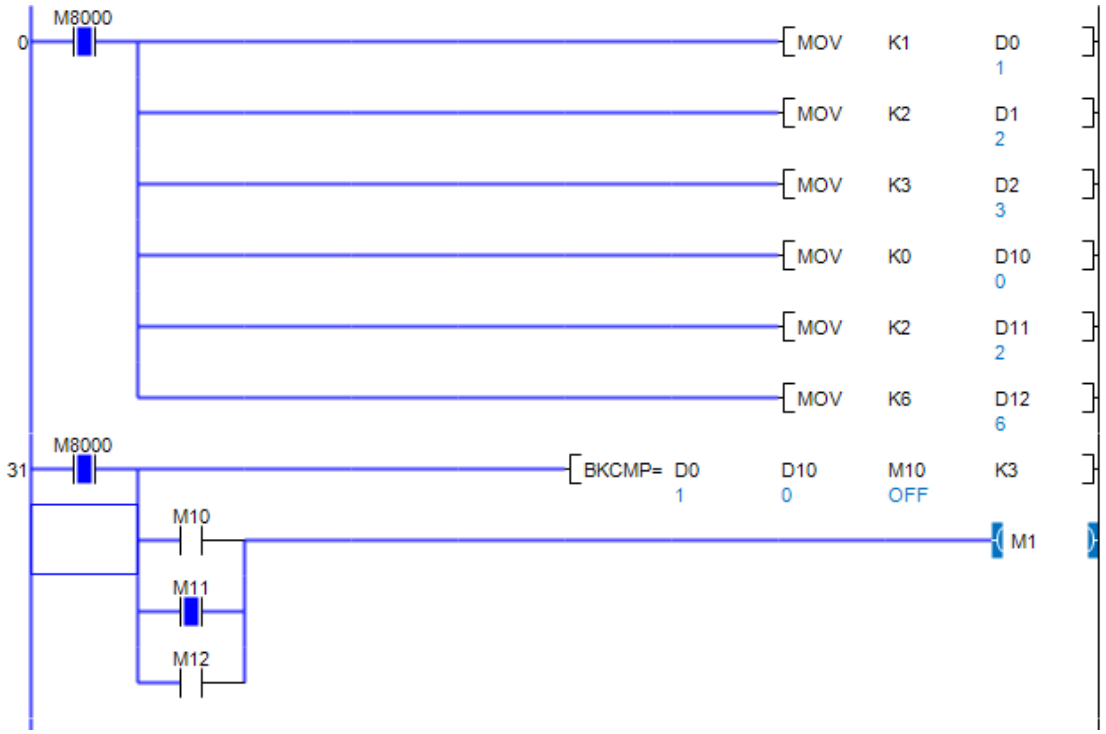
此外，比较结果(M10 开始的 4 点)全部为 ON(1)时，Y000 置 ON。



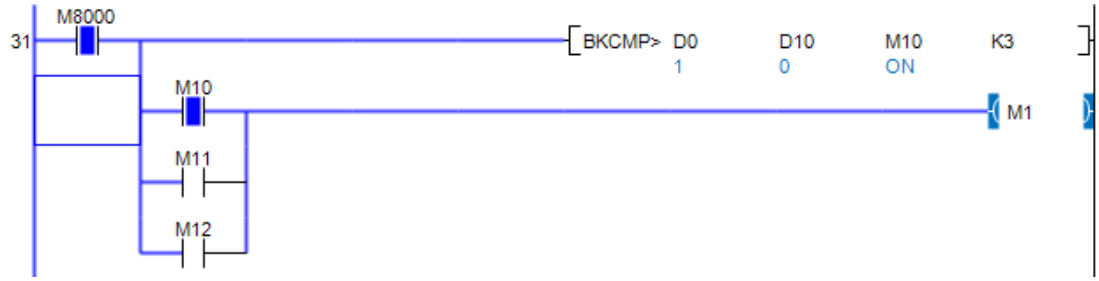
当X010为ON时，将常数K1000和D10开始的4点数据进行比较，然后将其结果保存到D0的b4～b7中的程序。



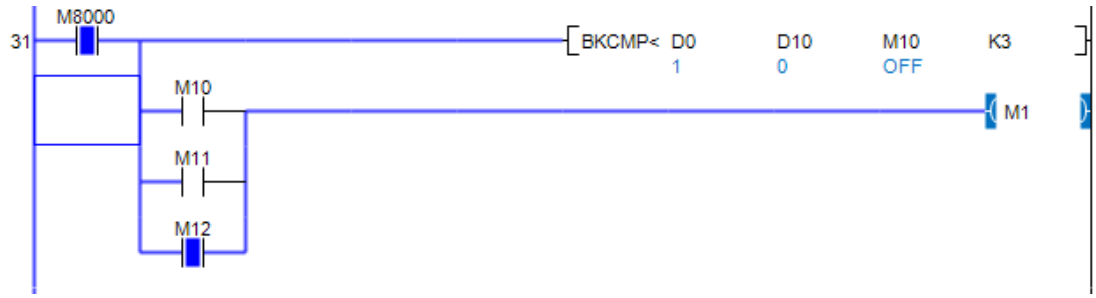
数据块等于比较



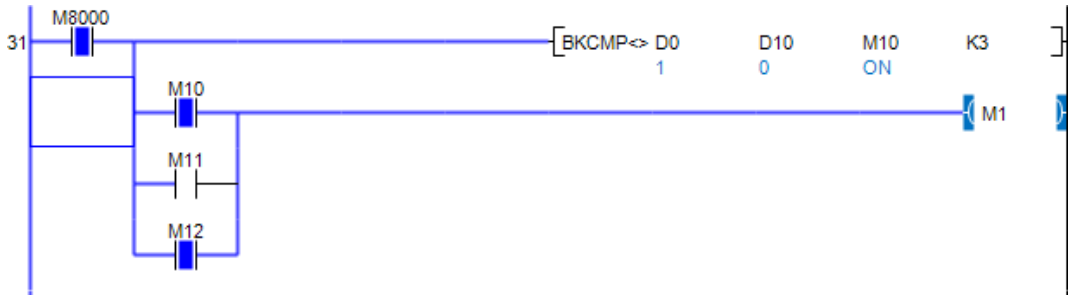
数据块大于指令



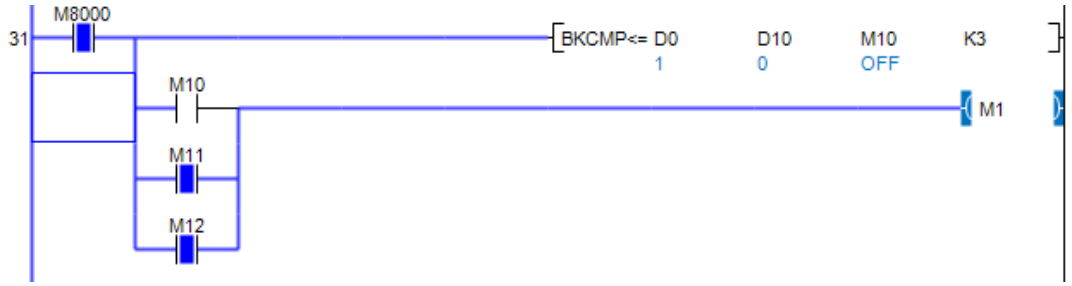
数据块小于比较



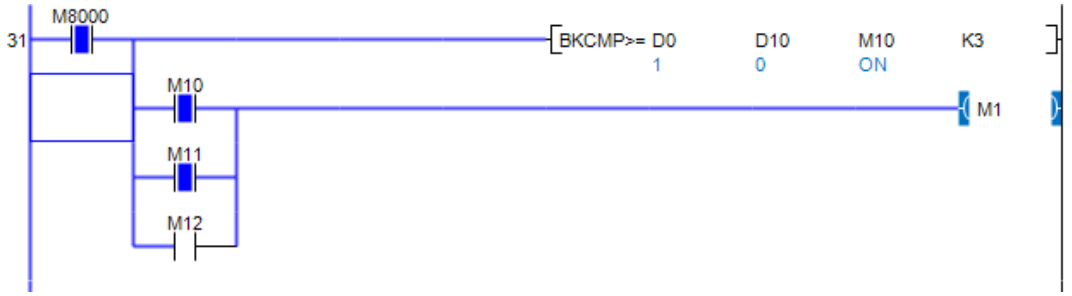
数据块不等于比较



数据块小于等于比较



数据块大于等于比较



4、注意事项

（1）指令比较结果如下

16 位指令	32 位指令	比较结果 ON 的条件	比较结果 OFF 的条件
BKCMP=	DBKCMP=	S1=S2	S1≠S2
BKCMP>	DBKCMP>	S1>S2	S1≤S2
BKCMP<	DBKCMP<	S1<S2	S1≥S2
BKCMP<>	DBKCMP<>	S1≠S2	S1=S2
BKCMP<=	DBKCMP<=	S1≤S2	S1>S2
BKCMP>=	DBKCMP>=	S1≥S2	S1<S2

（2）M8090 块比较信号的动作：数据块指令的比较结果都为 ON(1)时变为 ON。

3.17 字符串控制指令

3.17.1 $+指令(字符串的结合)

连接字符串与字符串的指令。

1、指令格式

16bit 7 步		32bit		指令格式
$+	$+P	\	\	$+ S1 S2 D

操作数的数据类型如下表

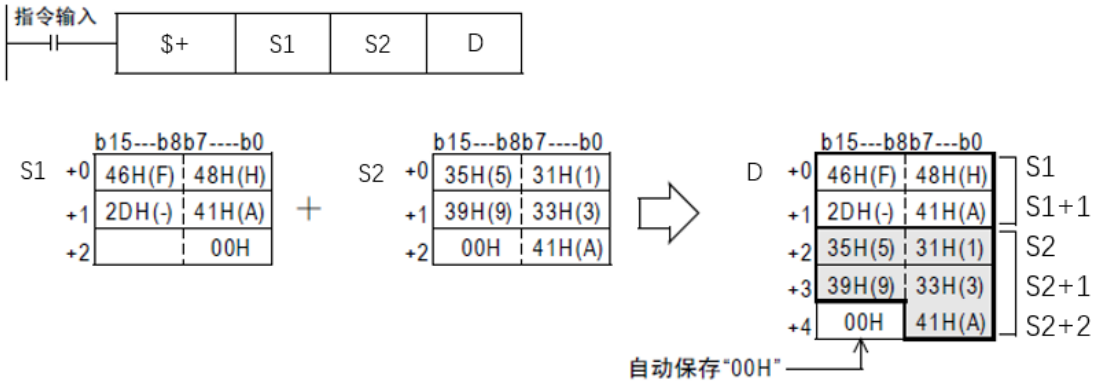
操作数	内容	类型
S1	保存连接源数据(字符串)的软元件起始编号， 或是被直接指定的字符串	字符串
S2	保存要连接的数据(字符串)的软元件起始编号， 或是被直接指定的字符串	字符串
D	保存连接后的数据(字符串)的软元件起始编号	字符串

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														●
D1														●
D2														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D1	●	●	●	●	●	●	●	●	●	●			●	
D2		●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明

在 S1 开始的字符串数据后面连接 S2 开始的字符串数据，然后保存到 D 以后的软元件中。S1 和 S2 的字符串，是指以字节为单位从被指定的软元件开始到检测到第一个[00H]的位置为止的数据。

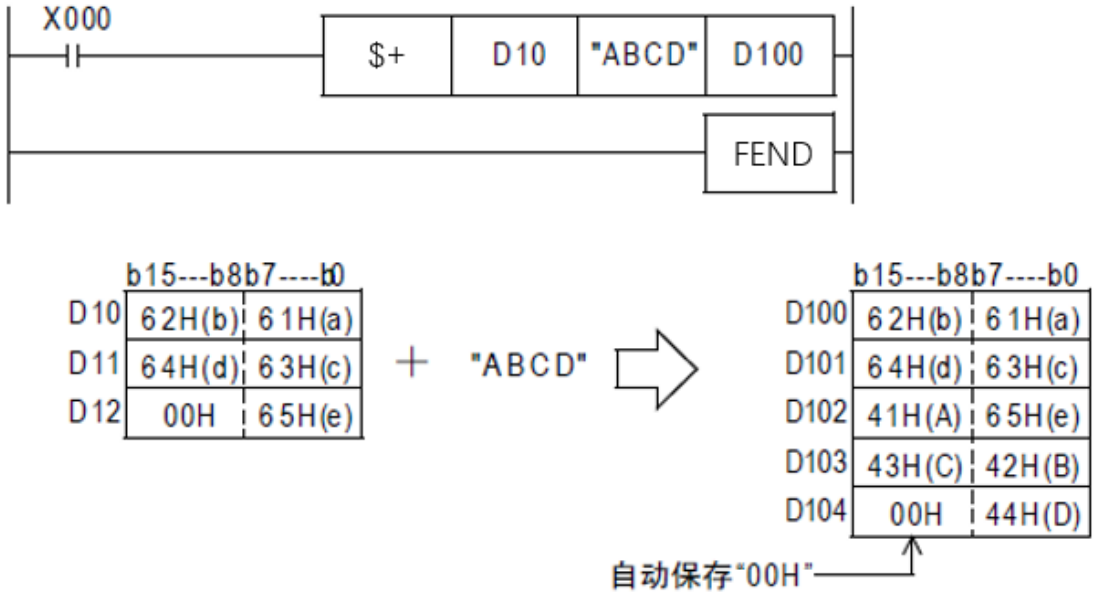


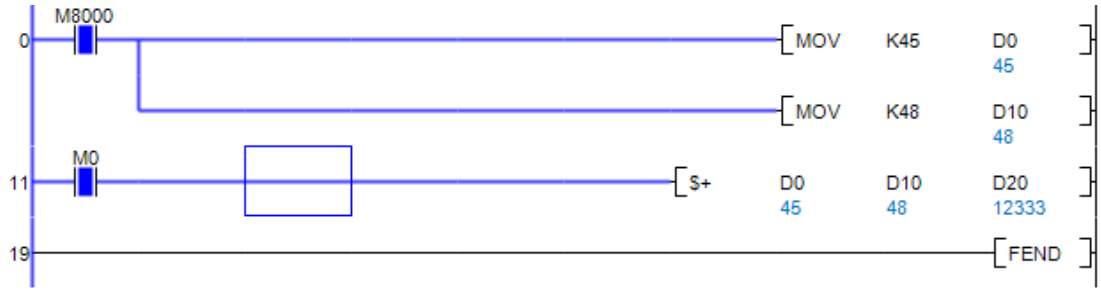
字符串的结合，就是指忽略 S1 中表示指定字符串末尾的“00H”，接着 S1 的最后字符连接 S2 中指定的字符串。此外，执行字符串的结合后会自动将“00H”附加在最后。

- 连接后的字符数为奇数时，在保存最后字符的软元件的高字节中保存“0000H”。
- 连接后的字符数为偶数时，在保存最后字符的软元件的下一个软元件中保存“0000H”。

3、程序举例

当X000为ON时，将D10～D12中保存的字符串(abcde)和字符串“ABCD”结合，然后保存到D100中的程序。





4、注意事项

（1）直接指定字符串时，可以指定(输入)的字符数最多为 32 个字符。 但是 S1 和 S2 中被指定了字软元件时，字符数没有限制。

（2）S1、S2 中任何一个的值从 00H 开始时，在 D 中保存“0000H”。

（3）以下情况报错 6706：

- D 中指定的软元件编号开始的软元件数，比保存所有已经结合的字符串所需的软元件数量更少时。(在所有的字符串和最终字符后面不能保存[00H])
- 当 S1 和 S2 中指定的保存字符串的软元件和 D 中指定的保存字符串的软元件编号重复时。
- 当 S1 和 S2 指定的软元件开始的相应软元件范围中[00H]未被设定时。

3.17.2 LEN 指令(检测字符串的长度)

检测出指定字符串的字符数(字节数)的指令。

1、指令格式

16bit 5 步		32bit		指令格式
LEN	LENP	\	\	LEN S D

操作数的数据类型如下表

操作数	内容	类型
S	保存要检测出字符数的字符串的软元件起始编号	字符串
D	保存已检测出的字符串的长度(字节数)的软元件编号	BIN16 位

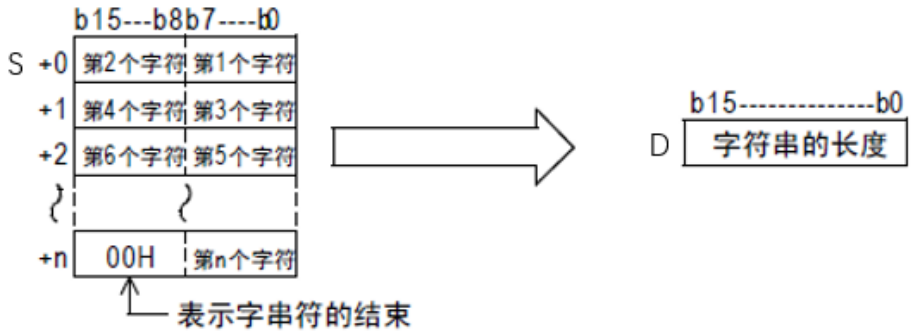
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														

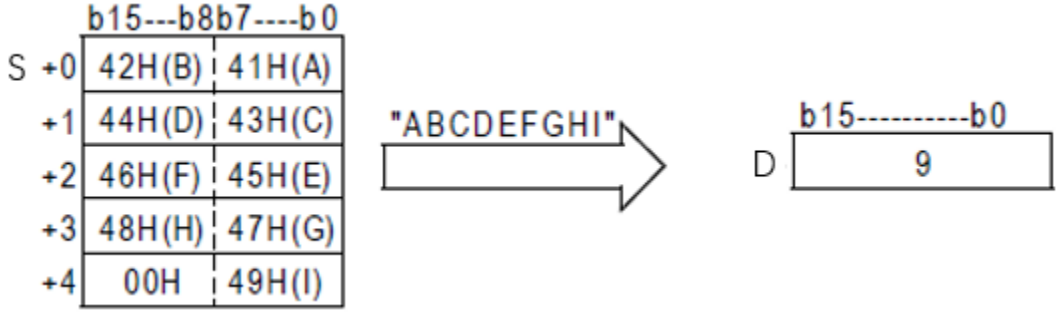
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明

检测出以 S 开头的字符串的长度，将字符串的长度保存到 D 中。S 以字节为单位，将从开始到第 1 个保存有[00H]的软元件编号为止的数据，作为字符串进行处理。

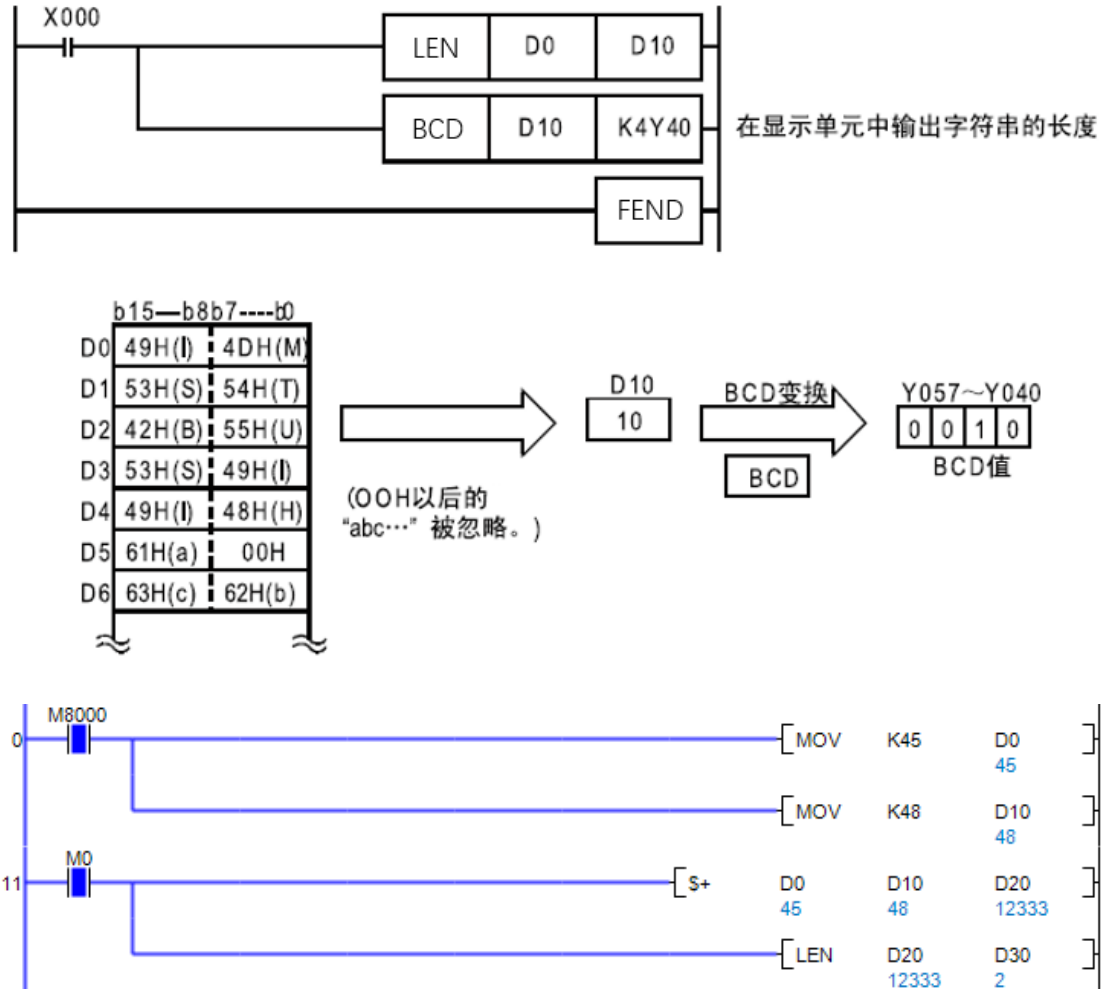


例如，如下所示的在 S 中保存“ABCDEFGHI”时， D 中保存 K9。



3、程序举例

当 X000 为 ON 时，将 D0 开始的字符串的长度，以 BCD4 位数形式输出到 Y040～Y057 中的程序。



4、注意事项

以下情况会报错 6706：

- 在 S 指定的软元件编号开始的相应软元件范围内没有设定[00H]时。
- 检测出字符数超过 32768 个时。

3.17.3 RIGHT 指令(从字符串的右侧开始读取)

从指定的字符串的右侧取出指定字符数的字符的指令。

1、指令格式

16bit 7 步		32bit		指令格式
RIGHT	RIGHTP	\	\	RIGHT S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存字符串的软元件起始编号	字符串
D	保存被取出的字符串的软元件起始编号	字符串
n	要取出的字符数	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	
n							●	●	●	●			●	

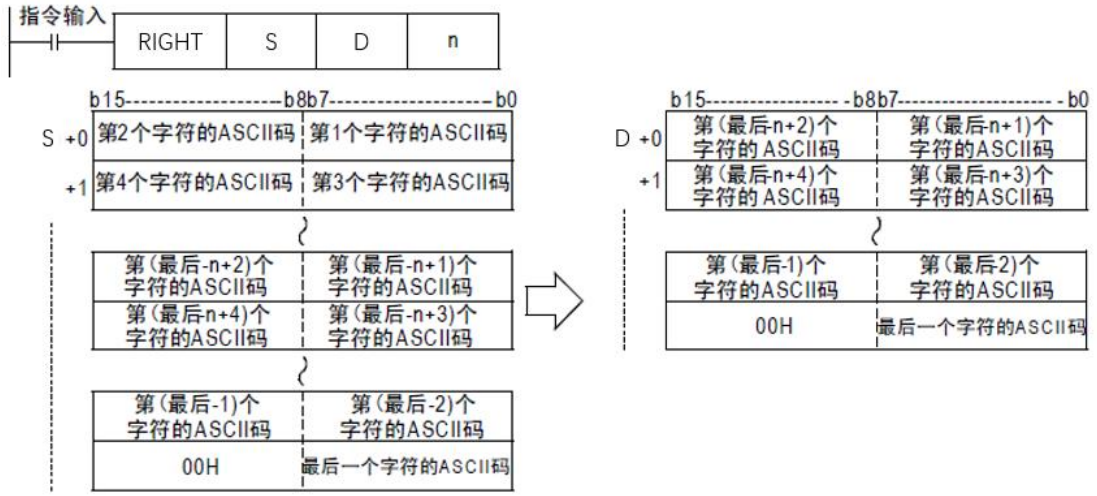
2、功能和动作说明

从 S 开始的软元件中保存的字符串数据的右侧(字符串的末尾)取出 n 个字符的数据，保存到 D 开始的软元件中。但是，n 中指定的字符数为“0”时，D 中保存 NULL 代码(0000H)。

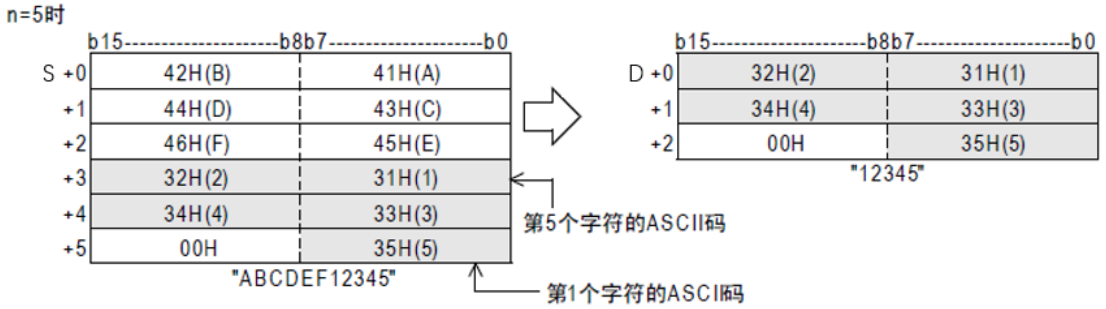
此外，取出字符串时，会在最后自动附加“00H”。

— 要取出的字符数为奇数时，在保存最后字符的软元件的高字节中保存“00H”。

— 要取出的字符数为偶数时， 在保存最后字符的软元件的下一个软元件中保“0000H”。



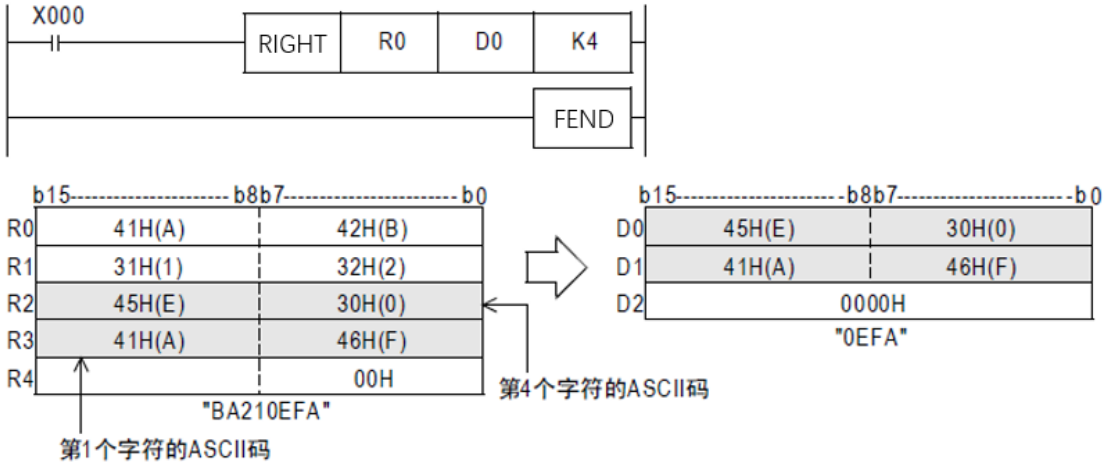
将 S 为起始的字符串数据“ABCDEF12345”的右 5 位字符存到 D 中：

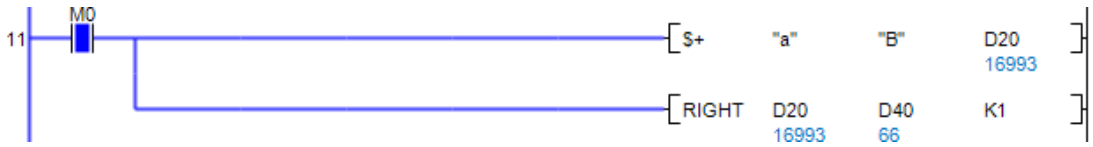


以 S 开头的字符串，就是指从被指定的软元件开始，以字节为单位，到检测出第 1 个[00H]的位置为止的数据。

3、程序举例

当X000为ON时，将R0开始的软元件中被保存的字符串数据的右侧起的4个字符数据，保存到D0开始的软元件中的程序。





“B”的 ASCII 码为 66

4、注意事项

以下情况会报错 6706：

- S 中指定的软元件开始的相应软元件范围内没有设定[00H]时。
- n 超出了 S 中指定的字符数时。

• D 中指定的软元件编号开始的软元件数，比保存已经取出的字符串(n 个字符)所需的软元件数更少时。(所有的字符串和最终字符后面不能保存[00H])
• n 为负值时。

3.17.4 LEFT 指令(从字符串的左侧开始读取)

从指定的字符串的左侧取出指定字符数的字符的指令。

1、指令格式

16bit 7 步		32bit		指令格式
LEFT	LEFTP	\	\	LEFT S D n

操作数的数据类型如下表

操作数	内容	类型
S	保存字符串的软元件起始编号	字符串
D	保存被取出的字符串的软元件起始编号	字符串
n	要取出的字符数	BIN16 位

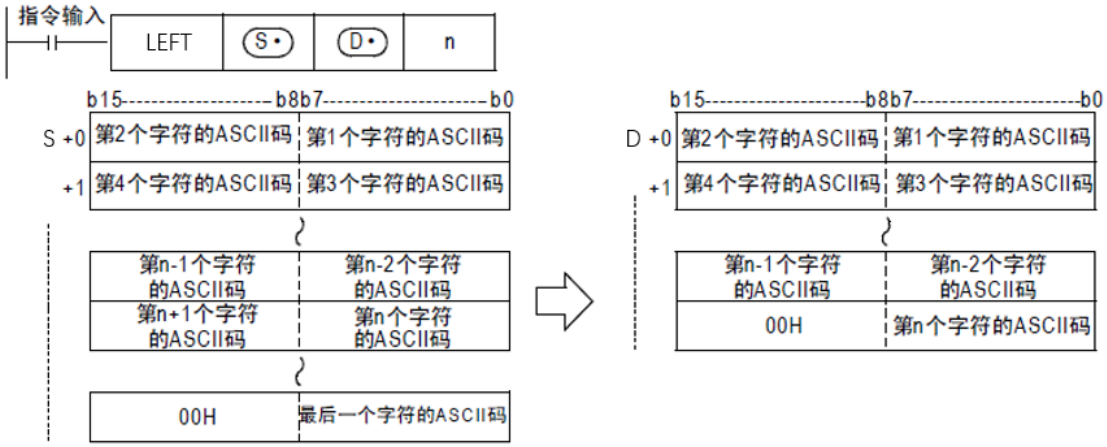
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	
n							●	●	●	●			●	

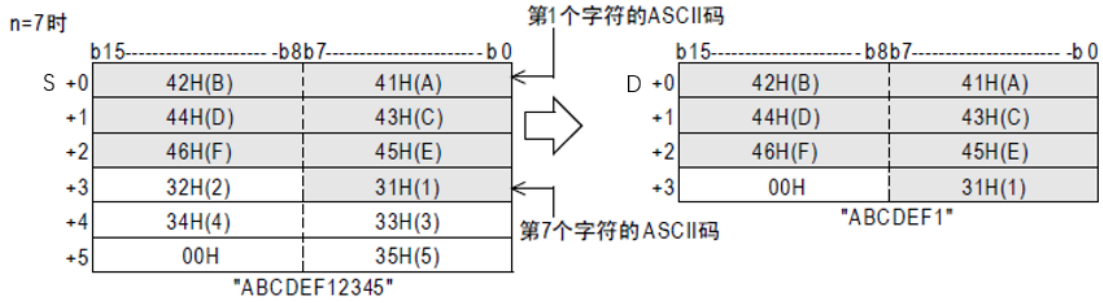
2、功能和动作说明

从 S 开始的软元件中保存的字符串数据的左侧(字符串的开头)取出 n 个字符的数据，保存到 D 开始的软元件中。但是，n 中指定的字符数为“0”时，D 中保存 NULL 代码(0000H)。
此外，取出字符串时，会在最后自动附加“00H”。

— 要取出的字符数为奇数时，在保存最后字符的软元件的高字节中保存“00H”。
— 要取出的字符数为偶数时， 在保存最后字符的软元件的下一个软元件中保“0000H”。



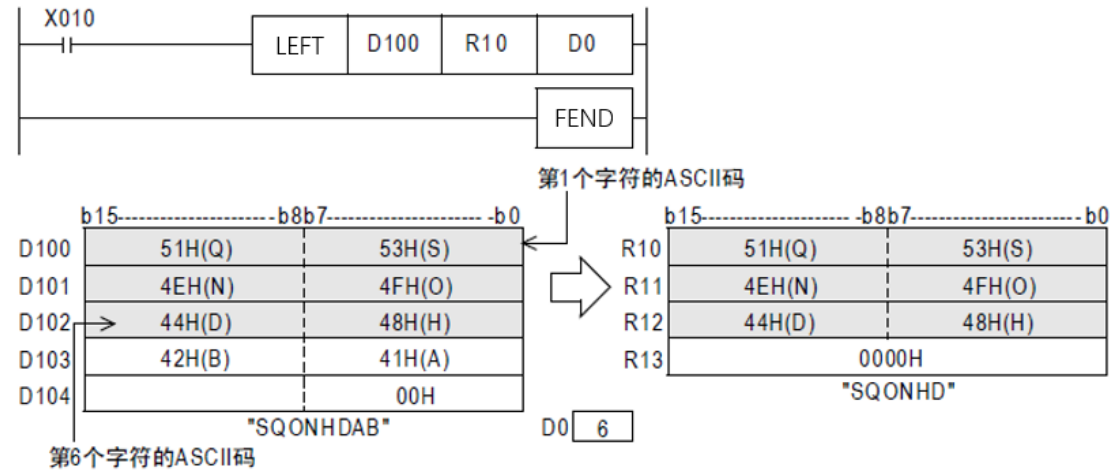
将 S 为起始的字符串数据“ABCDEF12345”的左 7 位字符存到 D 中：

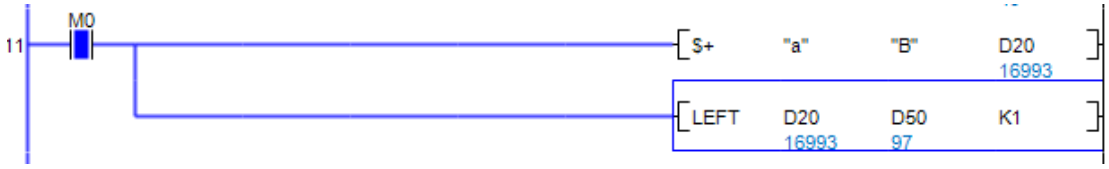


以 S 开头的字符串，就是指从被指定的软元件开始，以字节为单位，到检测出第 1 个[00H]的位置为止的数据。

3、程序举例

当X010为ON时，从D100开始的软元件中保存的字符串数据的左侧取出字符数据，字符数据个数为保存在D0中的值，将取出的字符数据保存到R10开始的软元件中的程序。





“a”的 ASCII 码为 97

4、注意事项

以下情况会报错 6706：

- S 中指定的软元件开始的相应软元件范围内没有设定[00H]时。
- n 超出了 S 中指定的字符数时。
- D 中指定的软元件编号开始的软元件数，比保存已经取出的字符串(n 个字符)所需的软元件数更少时。(所有的字符串和最终字符后面不能保存[00H])
- n 为负值时。

3.17.5 MIDR 指令从(字符串中任意取出)

取出指定的字符串中任意位置上的字符串的指令。

1、指令格式

16bit 7 步		32bit		指令格式
MIDR	MIDRP	\	\	MIDR S1 D S2

操作数的数据类型如下表

操作数	内容	类型

S1	保存字符串的软元件起始编号	字符串
D	保存被取出的字符串的软元件起始编号	字符串
S2	指定要取出的字符的起始位置以及字符数的软元件起始编号 S2：起始字符位置 S2+1：字符数	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
D														
S2														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	
S2	●	●	●	●	●	●	●	●	●	●			●	

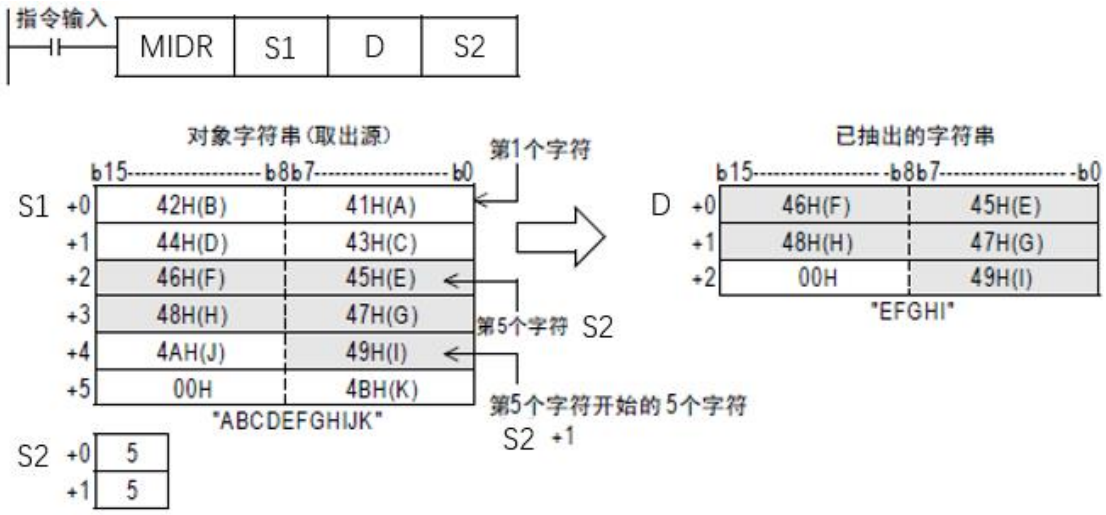
2、功能和动作说明

从 S1 开始的软元件中保存的字符串数据的左侧(字符串的开头)起第 S2 个字符开始，取出（S2+1）个字符的数据，保存到 D 开始的软元件中。

此外，取出字符串时，会在最后自动附加“00H”。

— 要取出的字符数（S2+1）为奇数时，在保存最后字符的软元件的高字节中保存“00H”。

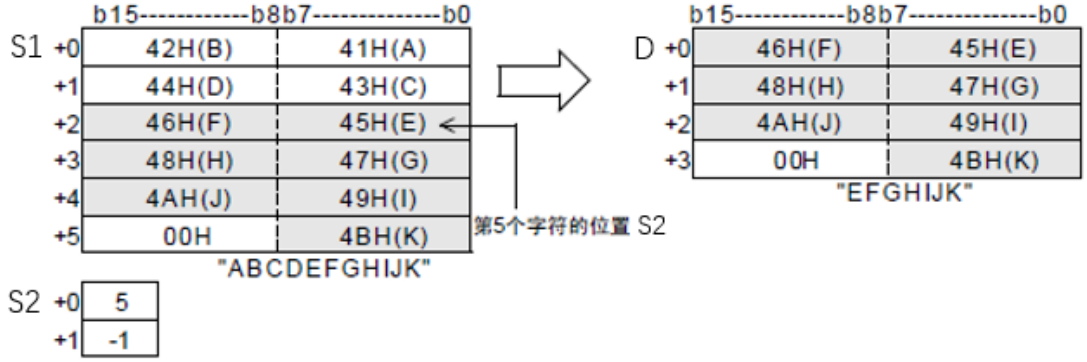
— 要取出的字符数(S2+1)为偶数时， 在保存最后字符的软元件的下一个软元件中保存“0000H”。



• S1 中指定的字符串(数据)，就是指从被指定的软元件开始到检测到第 1 个[00H]为止的数据。

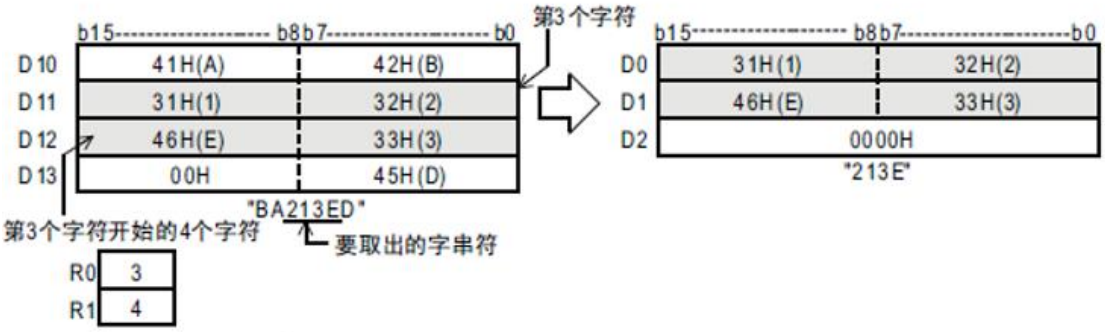
• S2+1 中指定的要取出的字符数为“0”时不执行处理。

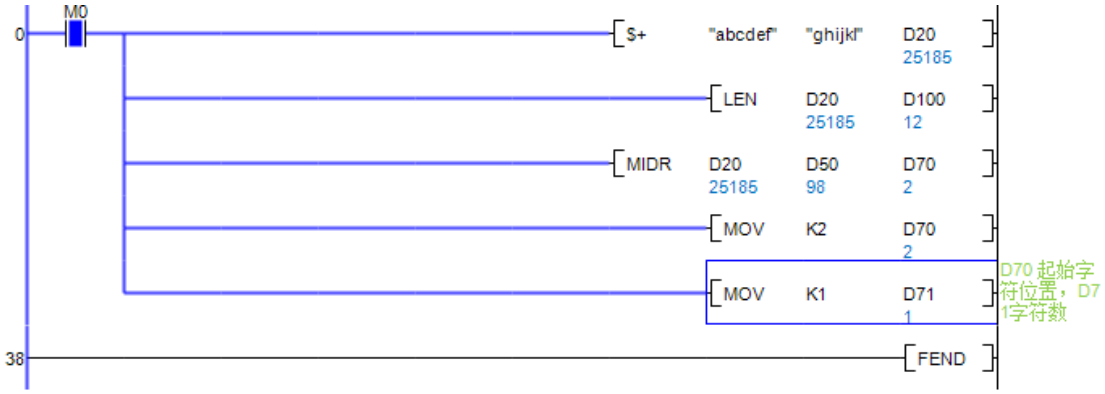
• S2+1 中指定的要取出的字符数为“-1”时，到 S1 指定的字符串的最终字符数据为止的内容被保存在中。



2、程序举例

当X000为ON时，从D10开始的软元件中保存的字符串数据的左侧开始数起，将第3个字符开始的4个字符的数据保存到D0开始的软元件中的程序。





4、注意事项

以下情况会报错 6706：

- S1 中指定的软元件开始的相应软元件范围内没有设定[00H]时。
- S2 的值超出了 S1 中指定的字符串的字符数时。
- D 开始的（S2+1）的字符数超出了 D 的软元件范围时。
- D 中指定的软元件编号开始的软元件数，比保存已取出的字符串 （S2+1）个字符所需的软元件数更少时。(所有的字符串和最终字符后面不能保存[00H])
- S2 为负值时。
- S2+1 为-2 以下的值时。
- S2+1 超出了 S1 的字符数时。

3.17.6 MIDW 指令(字符串的任意替换)

用指定的字符串中任意位置上的字符串去替换指定的字符串的指令。

1、指令格式

16bit 7 步		32bit		指令格式
MIDW	MIDWP	\	\	MIDW S1 D S2

操作数的数据类型如下表

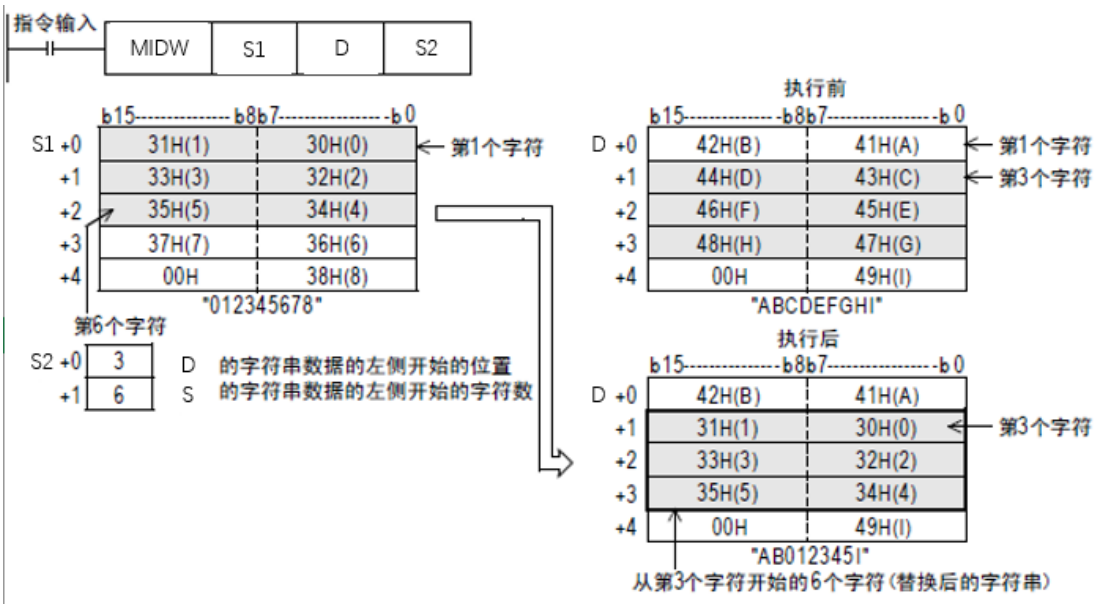
操作数	内容	类型
S1	保存字符串的软元件起始编号	字符串
D	保存替换后的字符串的软元件起始编号	字符串
S2	指定要替换的字符的起始位置以及字符数的软元件起始编号 S2：被替换的字符串的起始字符位置 S2+1：要替换的字符数	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														
D														
S2														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	
S2	●	●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明

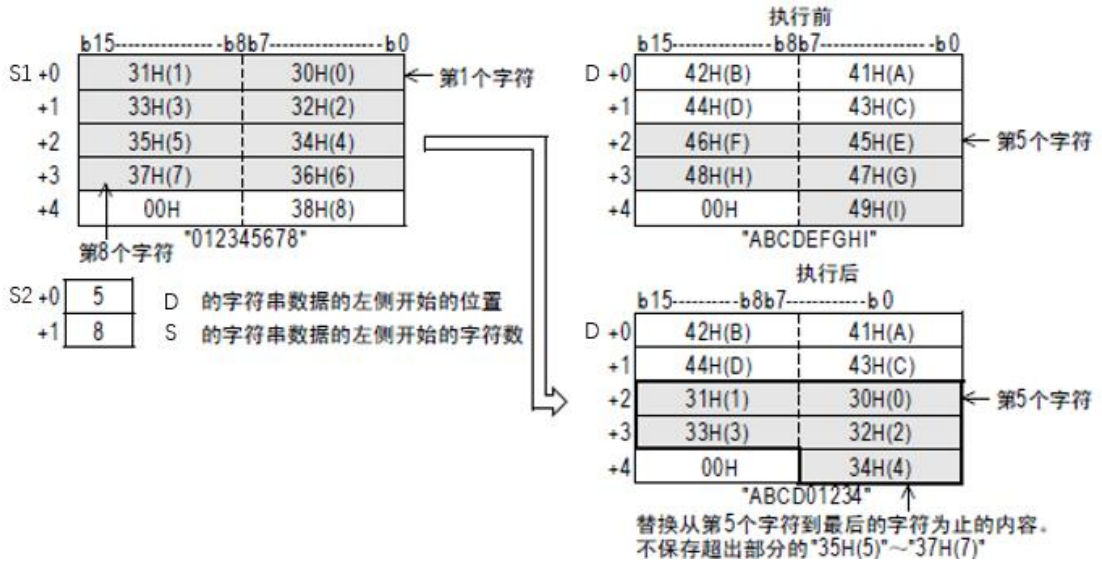
功能和动作说明针对 S1 起始的软元件中保存的字符串数据，将其左侧（ 字符串的开头）开始的字符数据，（ 字符数据数为 S2+1 中指定的字符数）保存到 D 起始软元件所保存的字符串中，保存位置从字符串数据左起数 S2 指定的位置开始。



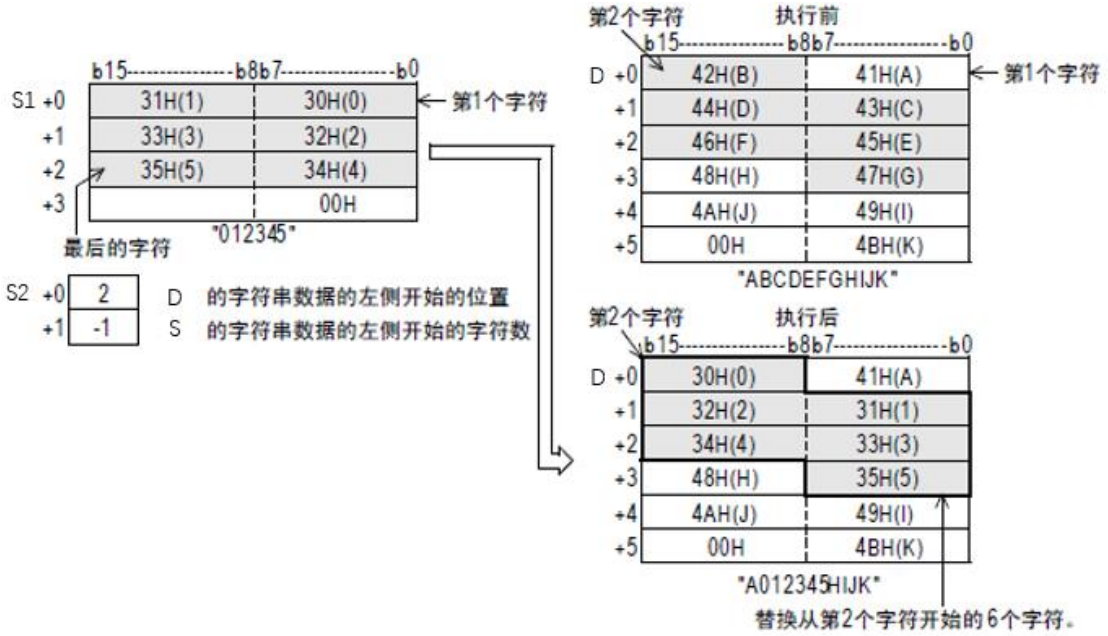
• S1 中 D 被指定的字符串(数据)，就是指从指定的软元件开始到检测开头到第 1 个[00H]为止的数据。

• S2+1 中指定的要替换的字符数为"0"时不执行处理。

• S2+1 中指定的要替换的字符数，一旦超出了 D 开始的字符串数据的最后字符时，则保存到最后字符为止的数据。

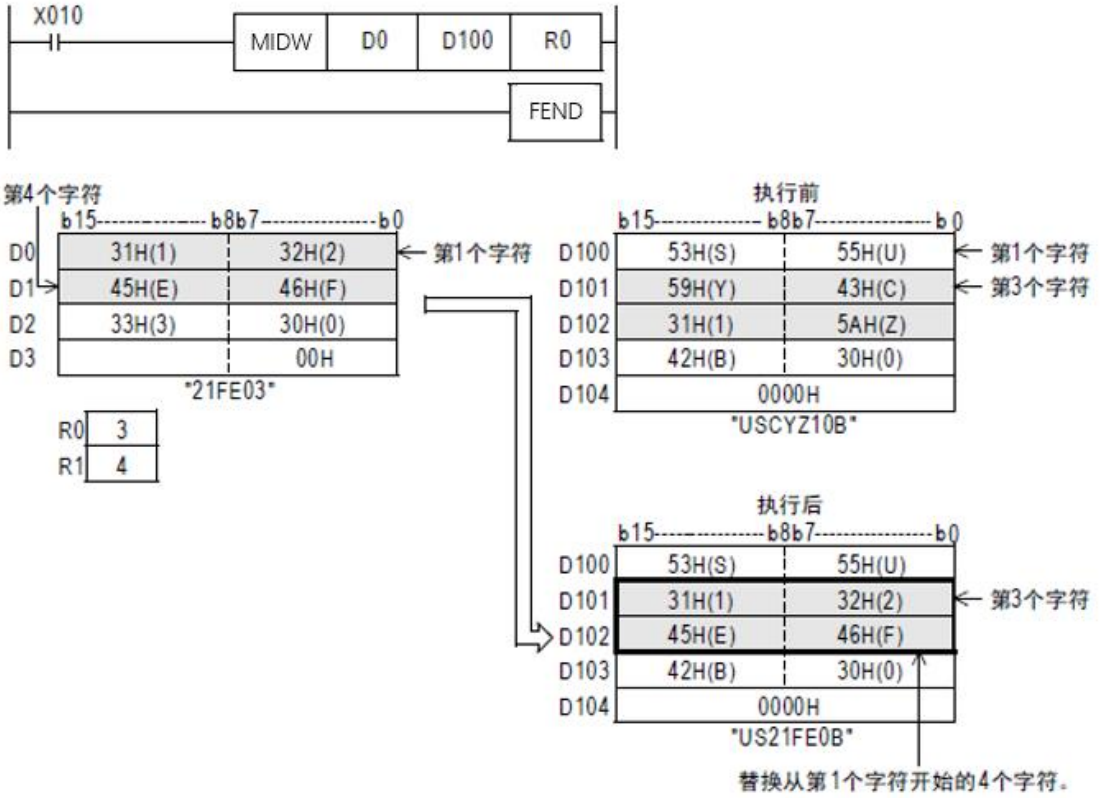


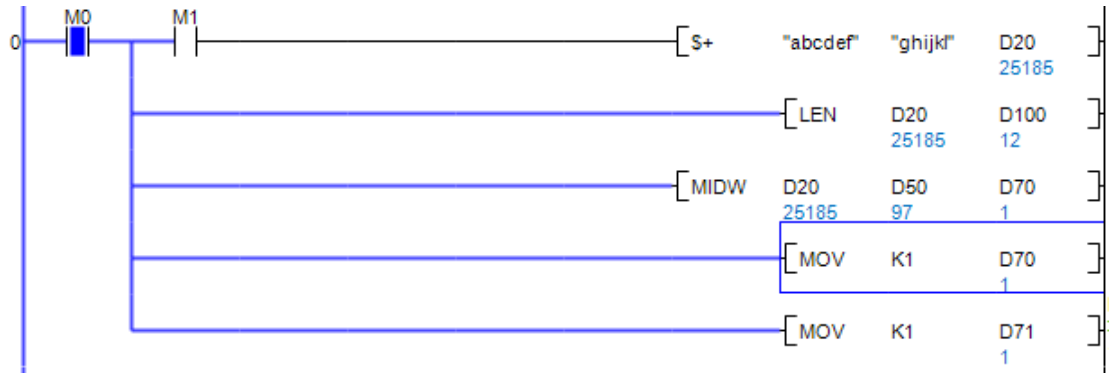
• S2+1 中指定的字符数为“-1”时，到 S1 指定的最终字符数据为止的内容都被保存到 D 指定的软元件以后中。



3、程序举例

程序为：当X010为ON时，将D0起始的软元件中保存的字符串数据中的4个字符，保存到D100起始的字符串数据中，保存位置从左起的第3个字符开始。





4、注意事项

以下情况报错 6706：

- S1 和 D 中指定的软元件开始的相应软元件范围内没有设定[00H]时。
- S2 的值超出了 D 的字符数时。
- S2 为负值时。
- S2+1 为-2 以下的值时。
- S2+1 超出了 S1 的字符数时。

3.17.7 INSTR 指令(字符串的检索)

从指定的字符串中检索指定字符串的指令。

1、指令格式

16bit 9 步		32bit		指令格式
INSTR	INSTRP	\	\	INSTR S1 S2 D n

操作数的数据类型如下表

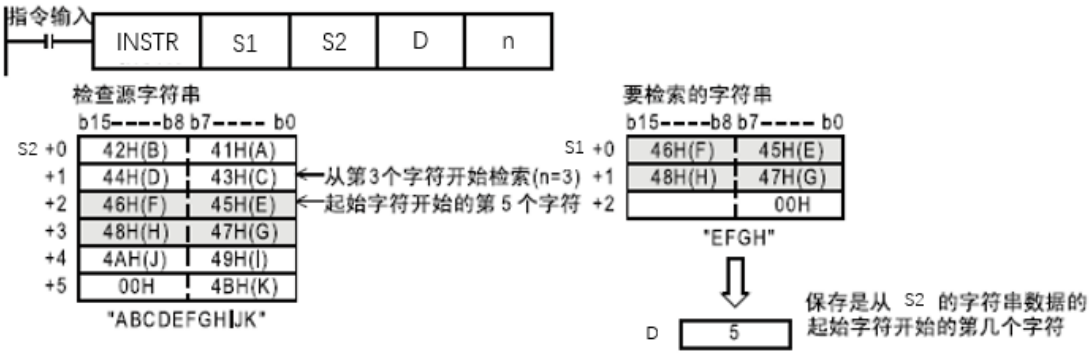
操作数	内容	类型
S1	保存要检索的字符串的软元件起始编号	字符串
S2	保存检索源字符串的软元件起始编号	字符串
D	保存检索结果的软元件起始编号	BIN16 位
n	开始检索的位置	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1														●
S2														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1					●	●	●	●	●	●			●	
S2					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●			●	

2、功能和动作说明

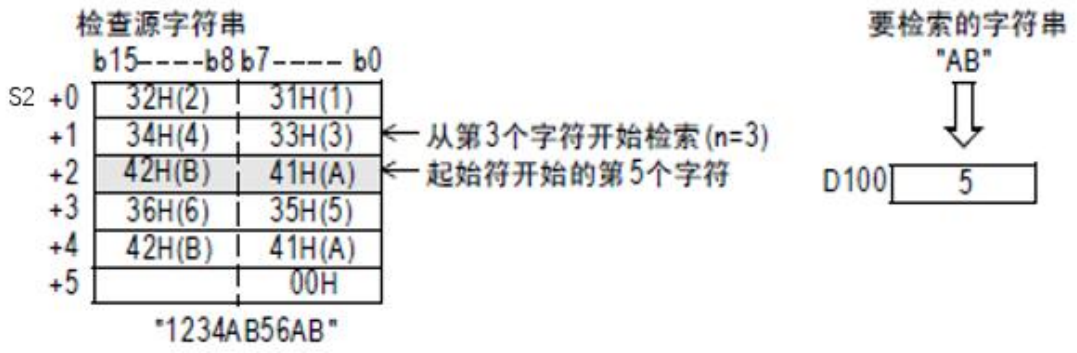
（1）从 S2 起始的软元件中保存的检索源字符串的左起(起始字符)第 n 个字符开始，检索与保存在 S1 起始的软元件中的字符串相同的字符串，然后将检索结果的字符串位置信息保存到 D 中。检索的结果，就是检索源字符串左起(起始字符)的第几个字符的字符位置信息。



（2）不存在一致的字符串时，在 D 中保存“0”。

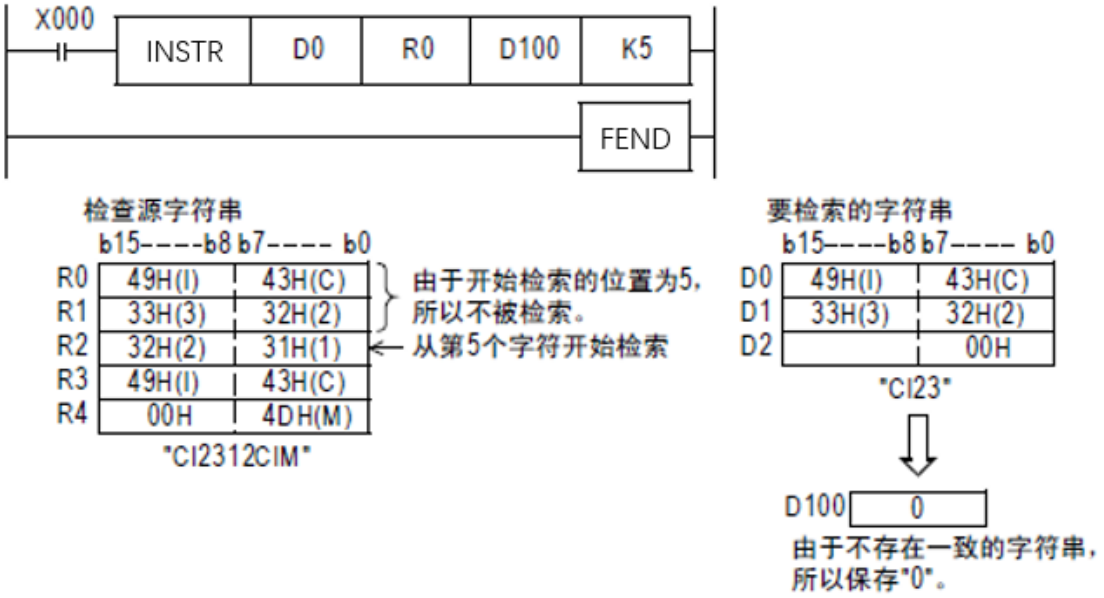
（3）开始检索的位置 n 为负数或是“0”不执行处理。

（4）在要检索的字符串 S1 中，可以直接指定字符串。

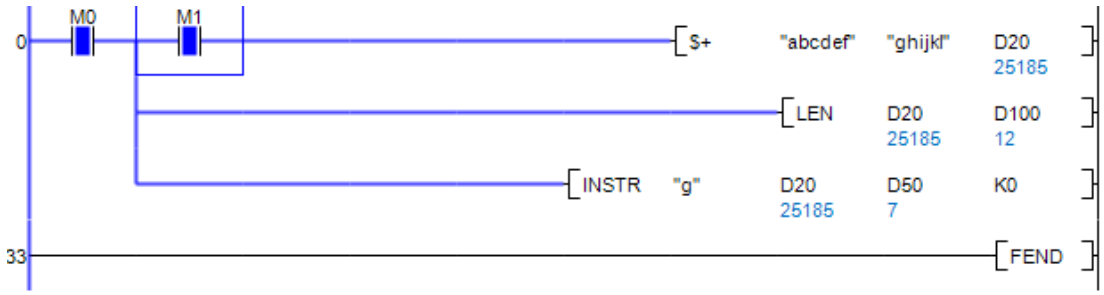


3、程序举例

当X000为ON时，从检索源字符串“CI2312CIM”(R0以后)的左起(起始字符)第5个字符开始检索要检索的字符串“CI23”(D0以后)，并将检索结果保存到D100中的程序。



使用$+指令生成字符串搜索“g”在第七位



4、注意事项

以下情况报错 6706：

- 开始检索的位置 n 超过了 S2 的字符数时。
- S1 开始的相应软元件的软元件范围内没有 00H(NULL)时。
- S2 开始的相应软元件的软元件范围内没有 00H(NULL)时。

3.17.8 $MOV 指令(字符串的传送)

传送字符串数据的指令。

1、指令格式

16bit 5 步		32bit		指令格式
$MOV	$MOVP	\	\	$MOV S D

操作数的数据类型如下表

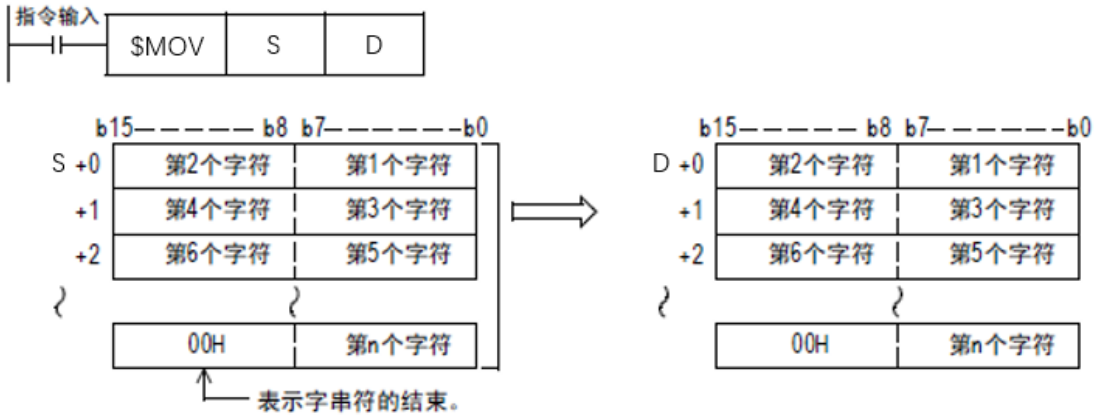
操作数	内容	类型
S	传送源中被直接指定的字符串(最大 32 个字符)，或是保存字符串的软元件起始编号	字符串
D	保存传送字符串的软元件起始编号	字符串

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														●
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	

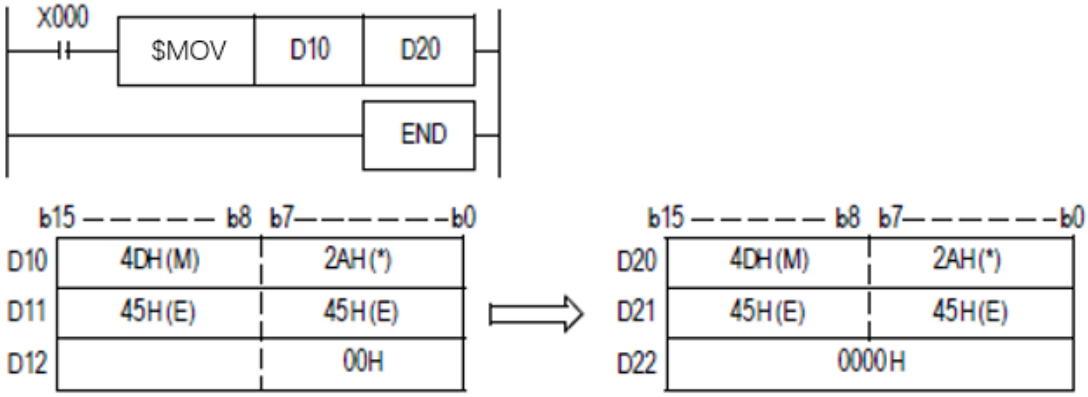
2、功能和动作说明

将 S 指定的软元件编号起始的软元件中保存的字符串数据，传送到 D 的软元件编号起始的软元件中。在传送字符串过程中，从 S 指定的软元件编号开始，一直到之后的软元件中，其高字节或是低字节中包含“00H”的软元件为止，都一次进行传送。



3、程序举例

当 X000 为 ON 时，将保存在 D10～D12 中的字符串数据传送到 D20～D22 中的程序。



4、注意事项

（1）S+n 的低字节中保存了“00H”时，D+n 的高字节、低字节中都保存“00H”。

（2）以下情况报错 6706：

- S 中指定的软元件编号～指定软元件的最终编号之间不存在“00H”时。
- 在 D 中指定的软元件编号～指定软元件的最终编号之间的点数中，不能保存所有指定的字符串时。

3.18 数据处理 3 指令

3.18.1 FDEL/FDEL2 指令(数据表的数据删除)

删除数据表格中任意数据的指令。

1、指令格式

16bit 7 步		32bit		指令格式
FDEL	FDELP	\	\	FDEL S D n

操作数的数据类型如下表

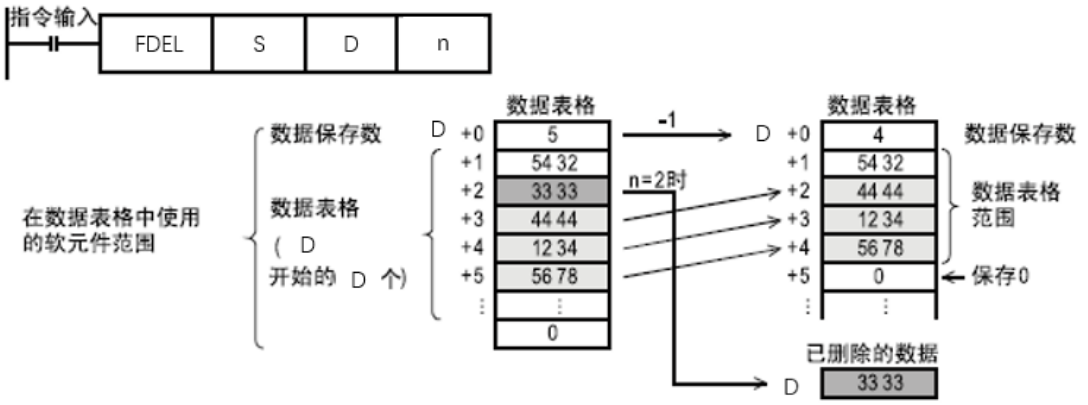
操作数	内容	类型
S	保存被删除数据的软元件编号	BIN16 位
D	数据表格的起始软元件编号	BIN16 位
n	要删除的数据的表格位置	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●				

2、功能和动作说明

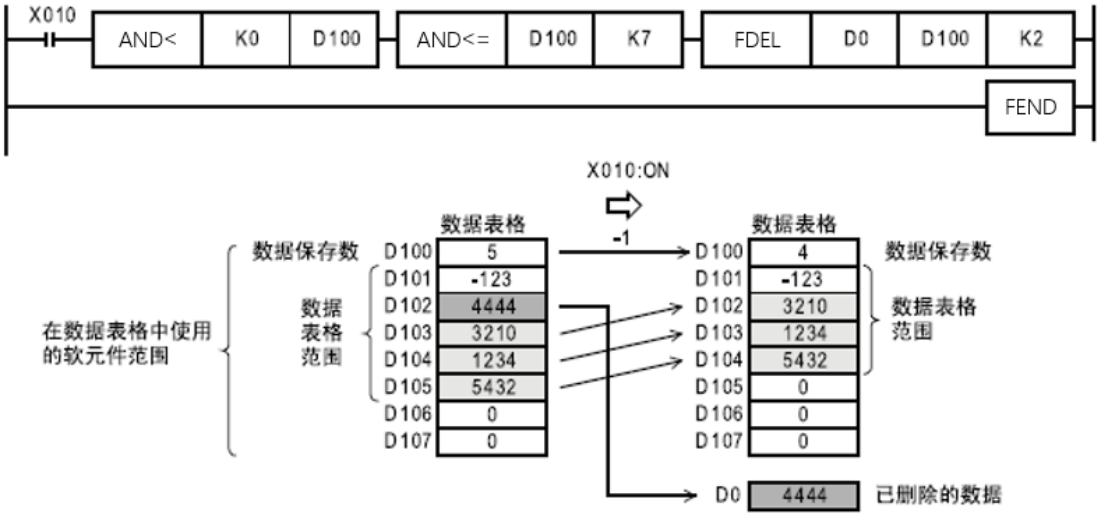
删除数据表格(D 起始)的第 n 个数据，将删除的数据保存到 S 中。数据表格的第 n+1 个开始的数据逐个向前靠拢，数据保存数减 1。



3、程序举例

当X010为ON时，删除D100～D105的数据表格中的第2个数据，将删除的数据保存到D0中的程序。

但是，当数据保存数为 0 时，请勿执行 FDEL 指令。(数据表格中使用的软元件范围为 D100～D107。)



4、注意事项

（1）数据表格的范围，为数据保存数 D 的下一个软元件(D+1)开始的 D 个。

（2）以下情况报错 6706：

- 要删除的数据的表格位置 n 比数据保存数更大时。
- n 的值超出了数据表格 D 的软元件范围时。
- 在 n≦0 的情况执行了指令时。

- 数据保存数 D 的值为 0 时。
- 数据表格的范围超出了相应的软元件范围时。

3.18.2 FINS/FINS2 指令(数据表的数据插入)

在数据表格中的任意位置处插入数据的指令。

1、指令格式

16bit 7 步		32bit		指令格式
FINS	FINSP	\	\	FINS S D n

操作数的数据类型如下表

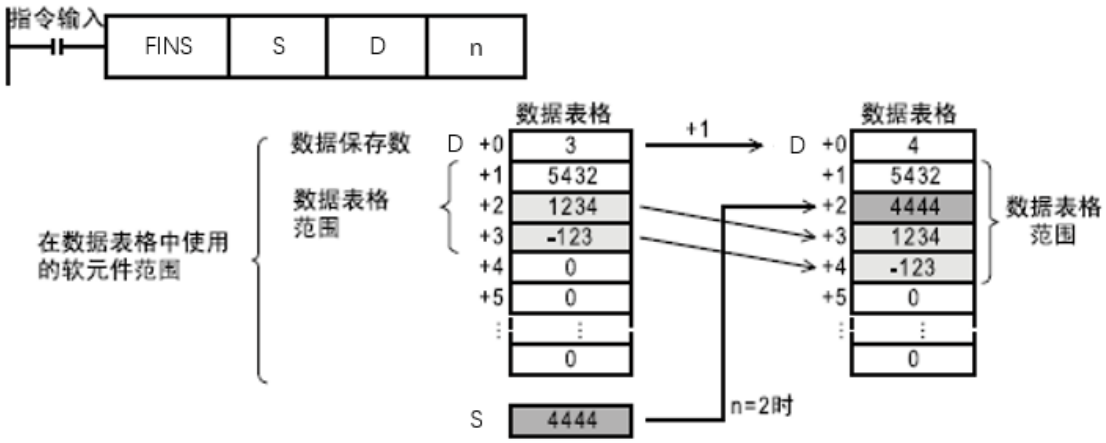
操作数	内容	类型
S	保存插入数据的软元件编号	BIN16 位
D	数据表格的起始软元件编号	BIN16 位
n	插入数据的表格位置	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	
D					●	●	●	●	●	●			●	
n							●	●	●	●				

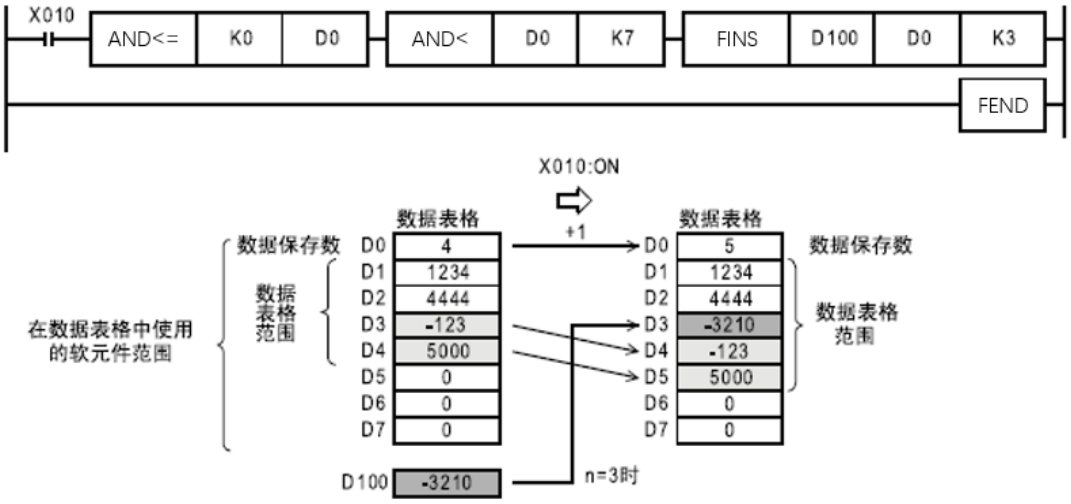
2、功能和动作说明

将 16 位数据 S 插入到数据表格(D 以后)的第 n 号中。数据表格的第 n 号以后的数据逐个后移，数据保存数加 1。



3、程序举例

当X010为ON时，在D0～D4的数据表格的第3号中插入D100的数据的程序。
但是，当数据保存数超过7时，不执行FINS指令。
（数据表格中使用的软元件范围为 D0～D7。）



4、注意事项

（1）数据表格的范围，为数据保存数 D 的下一个软元件(D+1)开始的 D 个。

（2）以下情况报错 6706：

- 要插入的数据的表格位置 n 比数据保存数+1 更大时。
- n 的值超出了数据表格 D 的软元件范围时。
- 在 n≦0 的情况执行了指令时。
- 数据表格的范围超出了相应的软元件范围时。

3.18.3 POP/POP2 指令(读取后入的数据)

该指令用于读出使用先入后出控制用的移位写入指令(SFWR)，写入最后的数据

1、指令格式

16bit 7 步		32bit		指令格式
POP	POPP	\	\	POP S D n

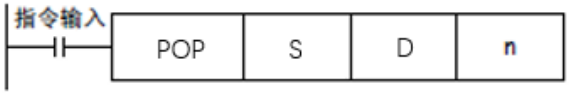
操作数的数据类型如下表

操作数	内容	类型
S	保存先入数据(包含指针数据)的起始软元件编号(保存数据的起始字软元件编号)	BIN16 位
D	保存后出的数据的软元件编号	BIN16 位
n	被保存的数据的点数(由于包含了指针数据，所以请设置为+1 后的值。)2≦n≦512	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S		●	●	●	●	●	●	●					●	
D		●	●	●	●	●	●	●	●	●	●	●	●	
n														

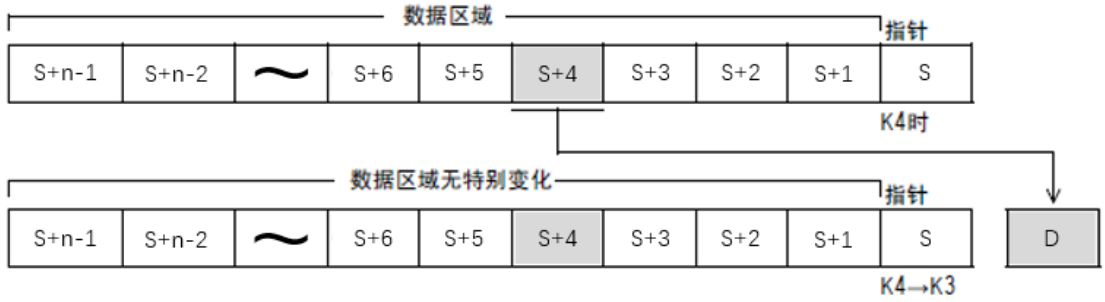
2、功能和动作说明



先入后出控制用数据

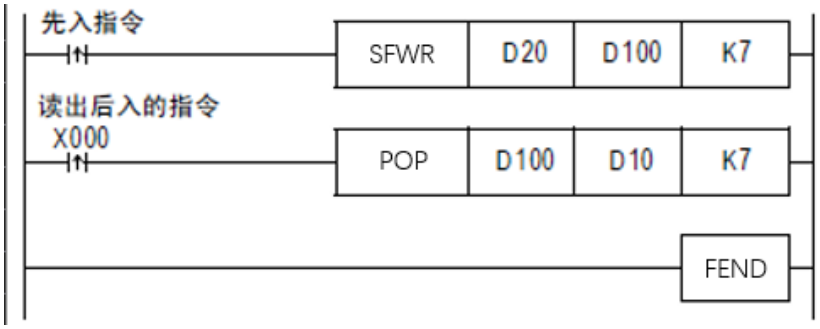
值	内容
S	指针数据（被保存的数据个数）
S+1	数据区域（使用唯一写入指令 SFWR 被先入的数据）
S+2	
...	
S+n-2	
S+n-1	

对于（S～S+n-1）的字软元件，每次执行指令时，读出（S +指针数据 S）的软元件保存到 D 中（使用先入先出控制用的移位写入指令（SFWR)写入的最后的数据读出到 D 中)，执行一次指针数据 S 的值减 1。



3、程序举例

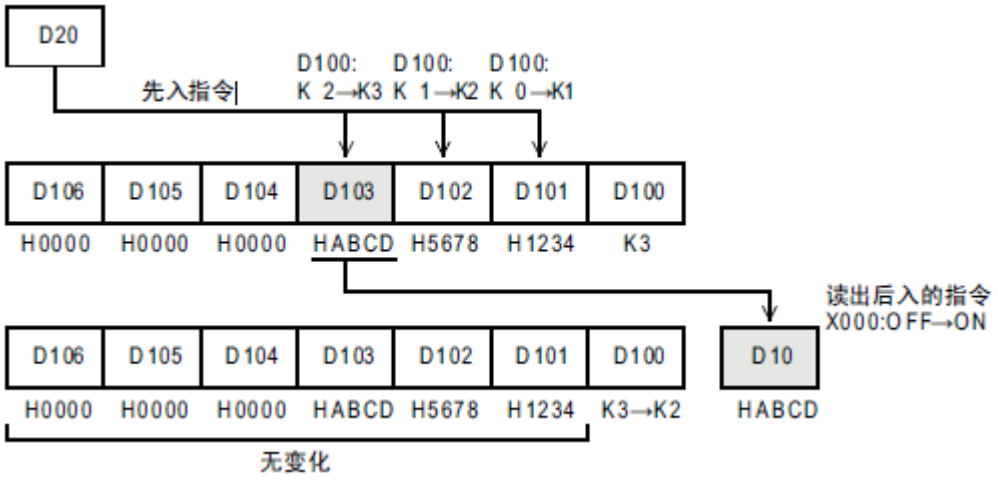
X000每次为ON时，对于D101～D106中被先输入的D20的值中，最后被保存的值都会保存到D10中，然后将数据保存数(指针D100)减1的程序。

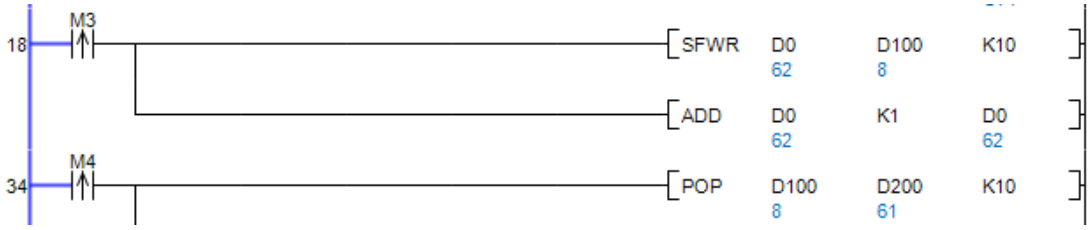


先输入的数据为下表中的内容时

指针	D100	K3
数据	D101	H1234
	D102	H5678
	D103	HABCD
	D104	H0000

	D105	H0000
	D106	H0000





4、注意事项

（1）M8020 的动作，当指针 S=0 时，执行指令后变为 ON，且不处理指令。

（2）指针的当前值为 1 时，S 中被写入 0，M8020 置 ON。

（3）以下情况报错 6706：

- S ＞ n-1 时。
- S ＜ 0 时。

3.18.4 SFR 指令(16 位数据的 n 位右移(带进位))

使字软元件中的 16 位向右移动 n 位的指令

1、指令格式

16bit 57 步		32bit		指令格式
SFR	SFRP	\	\	SFR D n

操作数的数据类型如下表

操作数	内容	类型
D	保存要移动的数据的起始字软元件编号	BIN16 位
n	移动的次数 0≦n≦15	BIN16 位

操作数的对象软元件如下表

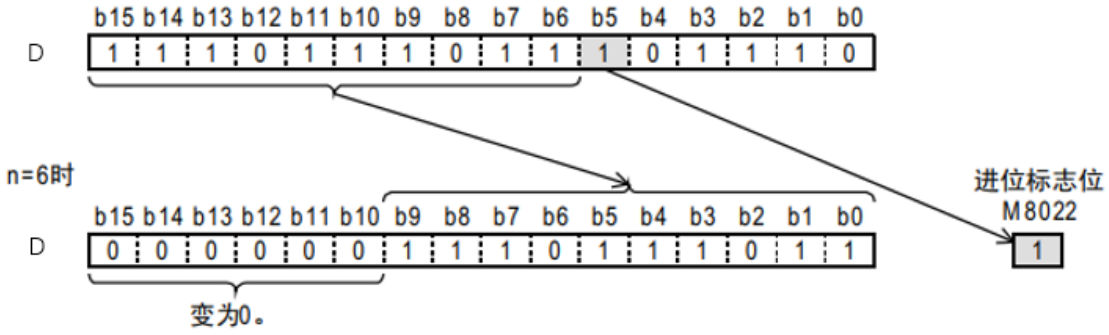
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●	●	●	●	
n	●	●	●	●	●	●	●	●	●	●	●	●		

2、功能和动作说明

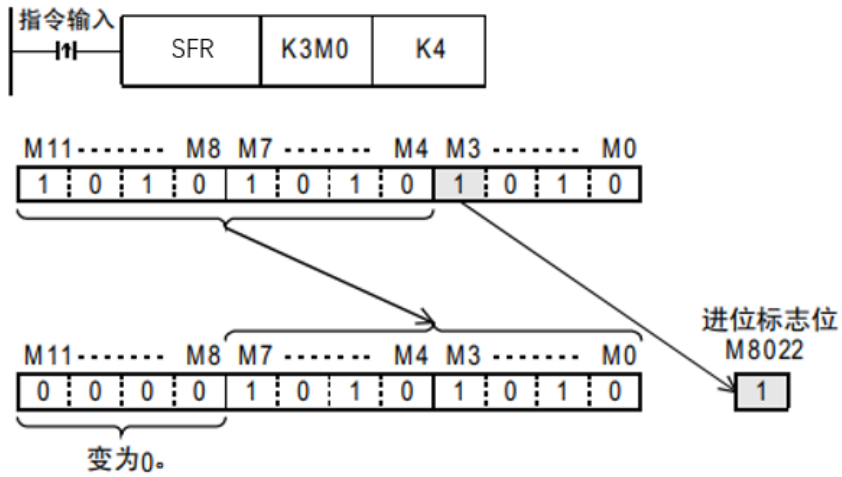
（1）字软元件 D 中的 16 位右移 n 位。n 指定 0～15 的数字。当 n 中指定了 16 以上的数值时，根据 n/16 的余数移动。例如，如 n=18 时，18/16=1 余 2，所以右移 2 位。

（2） 将字软元件 D 中的第 n 位(n-1 位)的 ON(1)/OFF(0)状态转移到进位标志 M8022 中。

（3） 最高位开始的 n 位变为 0。

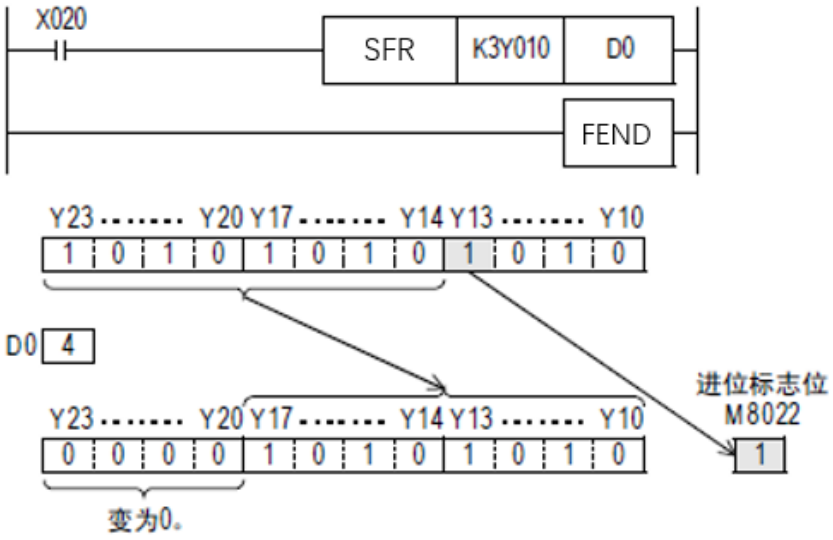


（4）通过位数指定来指定位软元件时，将指定位数(4×K□)的数据进行移位。



3、程序举例

当 X020 为 ON 时，将 Y010～Y023 的内容按照 D0 中指定的位数右移的程序。



4、注意事项

（1）M8022 进位标志的动作，根据移动 n-1 的状态 ON/OFF。

（2）n 为负值时报错 6706。

3.18.5 SFL 指令(16 位数据的 n 位左移(带进位))

使字软元件中的 16 位向左移动 n 位的指令

1、指令格式

16bit 5 步		32bit		指令格式
SFL	SFLP	\	\	SFL D n

操作数的数据类型如下表

操作数	内容	类型
D	保存要移动的数据的起始字软元件编号	BIN16 位
n	移动的次数 0≦n≦15	BIN16 位

操作数的对象软元件如下表

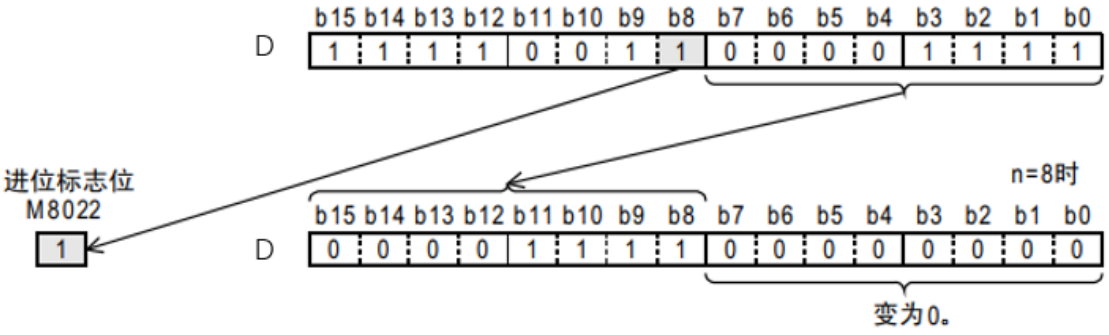
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
D														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
D		●	●	●	●	●	●	●	●	●	●	●	●	
n	●	●	●	●	●	●	●	●	●	●	●	●		

2、功能和动作说明

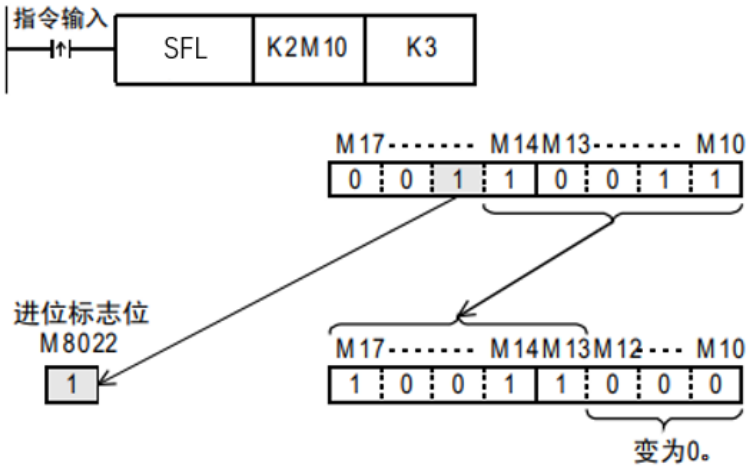
（1）字软元件 D 中的 16 位左移 n 位。n 指定 0～15 的数字。当 n 中指定了 16 以上的数值时，根据 n/16 的余数移动。例如，如 n=18 时，18/16=1 余 2，所以右移 2 位。

（2） 将字软元件 D 中的第 n+1 位(n 位)的 ON(1)/OFF(0)状态转移到进位标志 M8022 中。

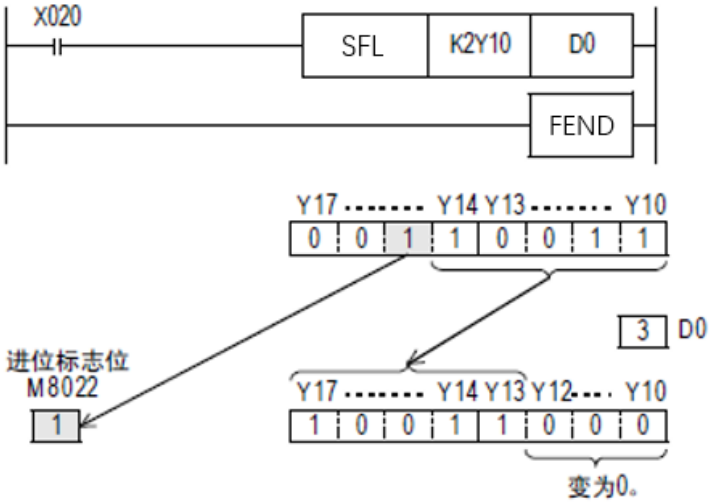
（3） 最低位开始的 n 位变为 0。



（4）通过位数指定来指定位软元件时，将指定位数(4×K□)的数据进行移位。



3、程序举例



4、注意事项

（1）M8022 进位标志的动作，根据移动 n 的状态 ON/OFF。

（2）n 为负值时报错 6706。

3.19 触点比较指令

提供了使用 LD、AND、OR 触点符号进行数据比较的指令

3.19.1 LD=、>、<、<>、<=、>=指令

执行数值的比较，当条件满足时使触点置 ON 的触点比较运算开始的指令。

1、指令格式

16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
LD=	\	LDD=	\	LDED=	\	LD= S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
LD>	\	LDD>	\	LDED>	\	LD> S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
LD<	\	LDD<	\	LDED<	\	LD< S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
LD<>	\	LDD<>	\	LDED<>	\	LD<> S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
LD<=	\	LDD<=	\	LEDD<=	\	LD<= S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
LD>=	\	LDD>=	\	LDED>=	\	LD=> S1 S2

操作数的数据类型如下表

操作数	内容	类型
S1	保存比较数据的软元件编号	BIN16/32 位
S2	保存比较数据的软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
S1											●	●		
S2											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

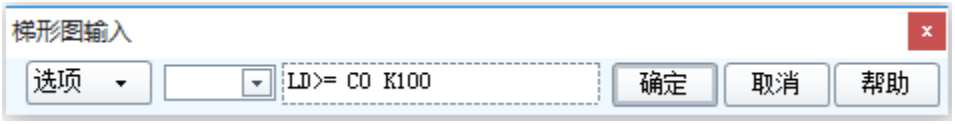
连接在母线上的触点比较指令。对 S1、S2 的内容进行 BIN 比较，根据其结果来控制触点的导通或是不导通。

16 位指令	32 位指令	32 位浮点指令	导通条件	不导通条件
LD=	LDD=	LDED=	S1=S2	S1≠S2
LD>	LDD>	LDED>	S1>S2	S1≤S2
LD<	LDD<	LDED<	S1<S2	S1≥S2
LD<>	LDD<>	LDED<>	S1≠S2	S1=S2
LD<=	LDD<=	LDED<=	S1≤S2	S1>S2
LD>=	LDD>=	LDED>=	S1≥S2	S1<S2

3、程序举例

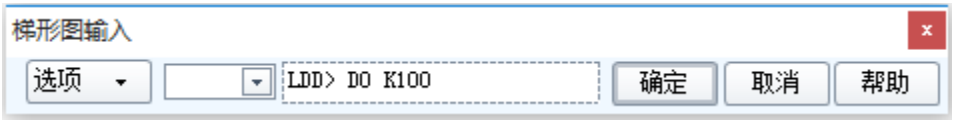
（1）16 位运算输入

数据比较指令，当(S1) > (S2)时接通



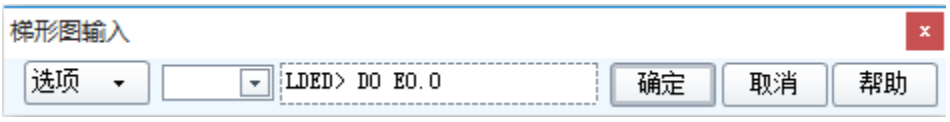
（2）32 位运算输入

数据比较指令，当(S1) > (S2)时接通

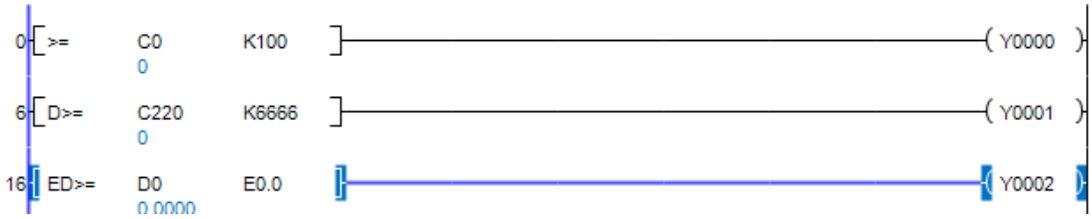


（3）32 位浮点运算输入

数据比较指令，当(S1) > (S2)时接通

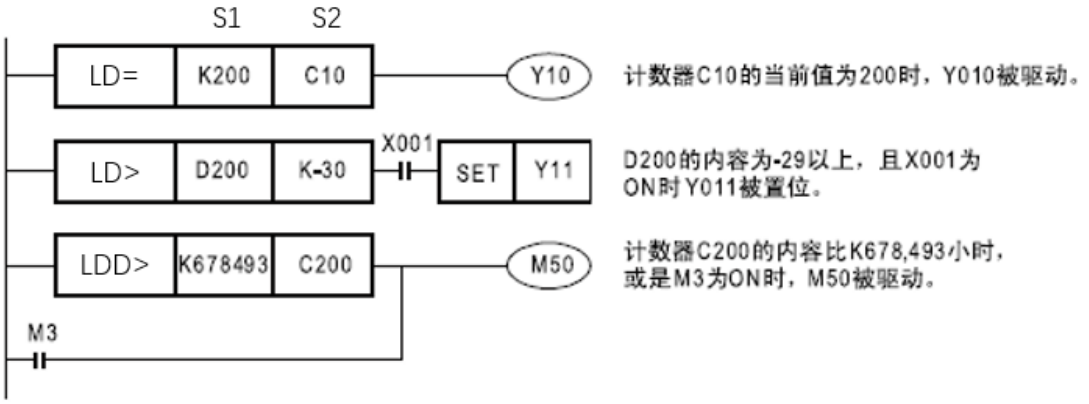


（4）实例 1



C0 值>=100 时，Y0 被驱动，C220 值>=6666 时 Y1 被驱动,D0>=0.0 的时候 Y2 被驱动。

实例 2



4、注意事项

（1）注意当比较源是 32 位计数器的时候选择 32 位指令。

（2）注意指令的书写方式。

3.19.2 AND=、>、<、<>、<=、>=指令

执行数值的比较，当条件满足时使触点置 ON 的触点比较运算开始的指令。

1、指令格式

16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
AND=	\	ANDD=	\	ANED=	\	AND= S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
AND>	\	ANDD>	\	ANED>	\	AND> S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
AND<	\	ANDD<	\	ANED<	\	AND< S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式

AND<>	\	ANDD<>	\	ANED<>	\	AND<> S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
AND<=	\	ANDD<=	\	ANED<=	\	AND<= S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
AND>=	\	ANDD>=	\	ANED>=	\	AND=> S1 S2

操作数的数据类型如下表

操作数	内容	类型
S1	保存比较数据的软元件编号	BIN16/32 位
S2	保存比较数据的软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

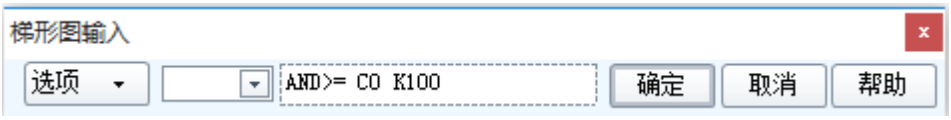
与其他触点串联的触点比较指令。对 S1、S2 的内容进行 BIN 比较，根据其结果来控制触点的导通或是不导通。

16 位指令	32 位指令	32 位浮点数	导通条件	不导通条件
AND=	ANDD=	ANED=	S1=S2	S1≠S2
AND>	ANDD>	ANED>	S1>S2	S1≤S2
AND<	ANDD<	ANED<	S1<S2	S1≥S2
AND<>	ANDD<>	ANED<>	S1≠S2	S1=S2
AND<=	ANDD<=	ANED<=	S1≤S2	S1>S2
AND>=	ANDD>=	ANED>=	S1≥S2	S1<S2

3、程序举例

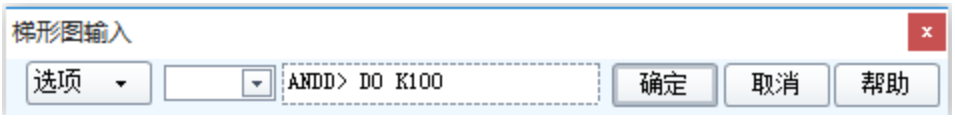
（1）16 位运算输入

数据比较指令，当(S1) > (S2)时接通

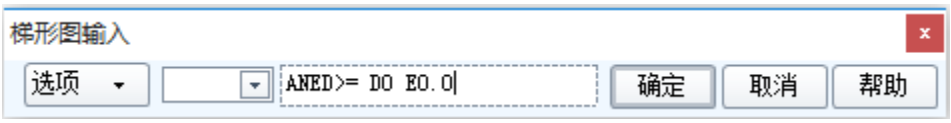


（2）32 位运算输入

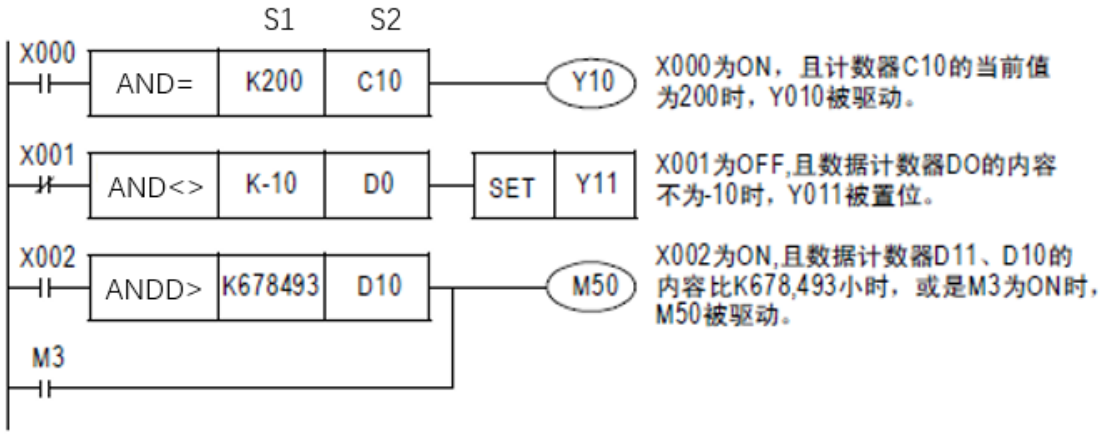
数据比较指令，当(S1) > (S2)时接通



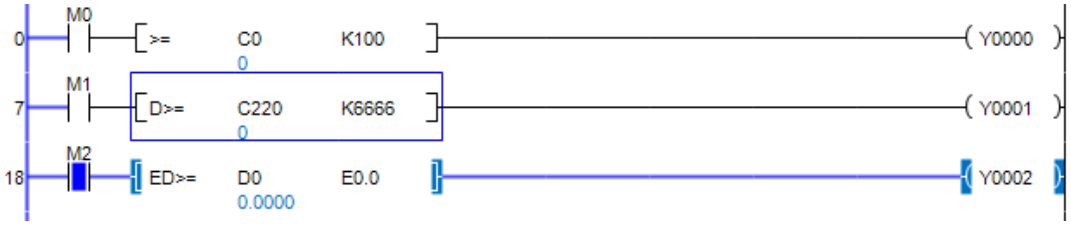
（3）32 位浮点数运算输入



（4）实例 1



实例 2



M0 为 ON 且 C0 值>=100 时，Y0 被驱动；M1 为 ON 且 C220 值>=6666 时 Y1 被驱动,M2 为 ON 的时候 D0 >= 0.0 时，Y2 被驱动。

4、注意事项

（1）注意当比较源是 32 位计数器的时候选择 32 位指令。

（2）注意指令的书写方式。

3.19.3 OR=、>、<、<>、<=、>=指令

执行数值的比较，当条件满足时使触点置 ON 的触点比较运算开始的指令。

1、指令格式

16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
OR=	\	ORD=	\	ORED=	\	OR= S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
OR>	\	ORD>	\	ORED>	\	OR> S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
OR<	\	ORD<	\	ORED<	\	OR< S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
OR<>	\	ORD<>	\	ORED<>	\	OR<> S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
OR<=	\	ORD<=	\	ORED<=	\	OR<= S1 S2
16bit 5 步		32bit 9 步		32bit float 9 步		指令格式
OR>=	\	ORD>=	\	ORED>=	\	OR=> S1 S2

操作数的数据类型如下表

操作数	内容	类型

S1	保存比较数据的软元件编号	BIN16/32 位
S2	保存比较数据的软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“ ”
S1											●	●		
S2											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

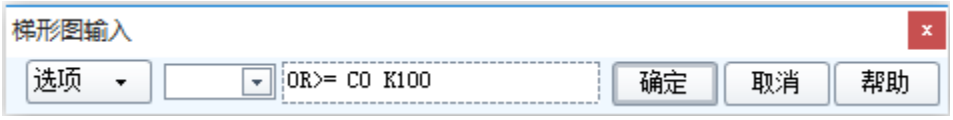
与其他触点并联的触点比较指令。对 S1、S2 的内容进行 BIN 比较，根据其结果来控制触点的导通或是不导通。

16 位指令	32 位指令	32 位浮点数	导通条件	不导通条件
OR=	ORD=	ORED=	S1=S2	S1≠S2
OR>	ORD>	ORED>	S1>S2	S1≤S2
OR<	ORD<	ORED<	S1<S2	S1≥S2
OR<>	ORD<>	ORED<>	S1≠S2	S1=S2
OR<=	ORD<=	ORED<=	S1≤S2	S1>S2
OR>=	ORD>=	ORED>=	S1≥S2	S1<S2

3、程序举例

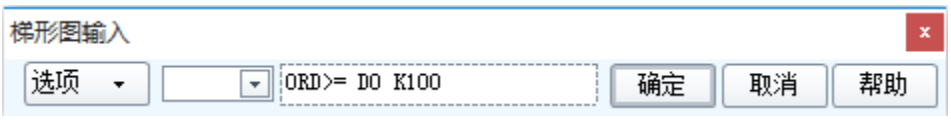
（1）16 位运算输入

数据比较指令，当(S1) > (S2)时接通



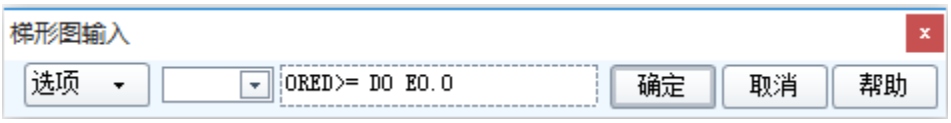
（2）32 位运算输入

数据比较指令，当(S1) > (S2)时接通

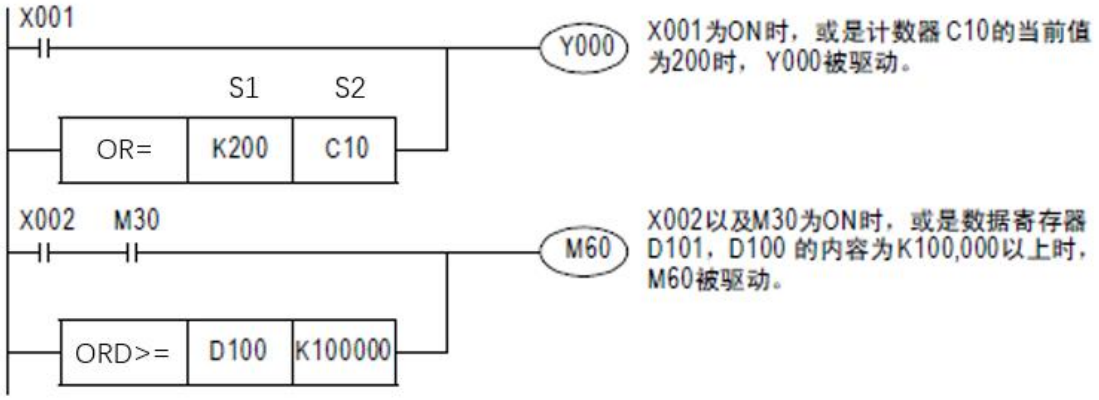


（3）32 位浮点数输入

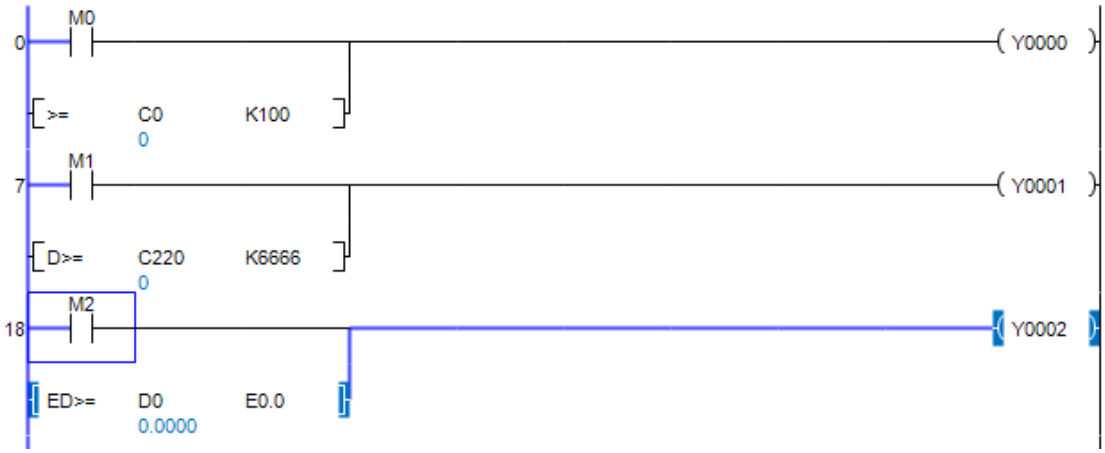
数据比较指令，当(S1) > (S2)时接通



（4）实例 1



实例 2



M0 为 ON 或 C0 值>=100 时，Y0 被驱动；M1 为 ON 或 C220 值>=6666 时 Y1 被驱动,M2 为 ON 或 D0>=0.0 的时，Y2 被驱动。

4、注意事项

（1）注意当比较源是 32 位计数器的时候选择 32 位指令。

（2）注意指令的书写方式。

3.20 数据表处理指令

3.20.1 LIMIT 指令(上限限位控制)

设置输入数值的上限值/下限值然后输出的指令。

1、指令格式

16bit 9 步		32bit 17 步		指令格式
LIMIT	LIMITP	DLIMIT	DLIMITP	LIMIT S1 S2 S3 D

操作数的数据类型如下表

操作数	内容	类型
S1	下限限位值(最小输出界限值)	BIN16/32 位
S2	上限限位值(最大输出界限值)	BIN16/32 位
S3	需要通过上下限限位控制的输入值	BIN16/32 位
D	保存已经过上下限限位控制的输出值的软元件起始编号	BIN16/32 位

操作数的对象软元件如下表

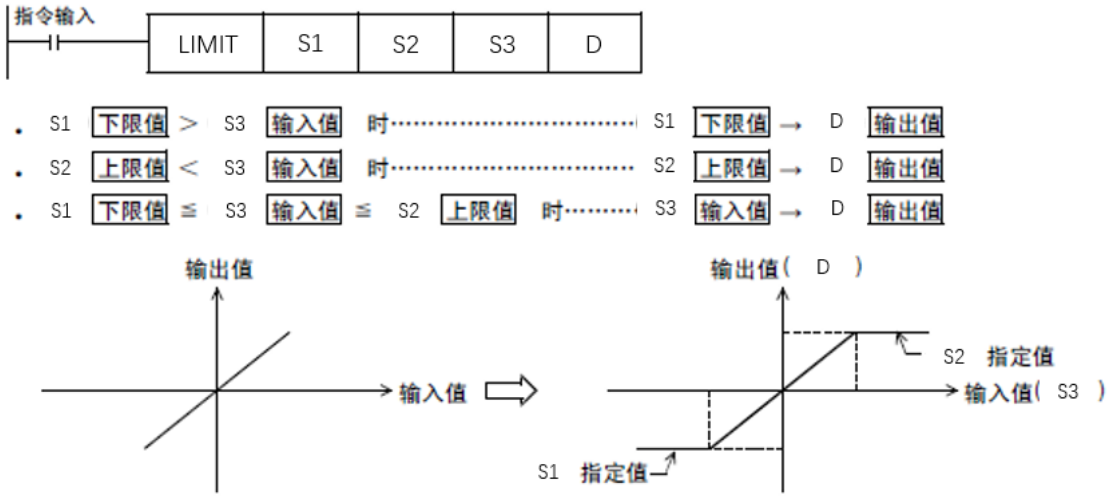
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
S3														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●			●	
S2	●	●	●	●	●	●	●	●	●	●			●	

S3	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明

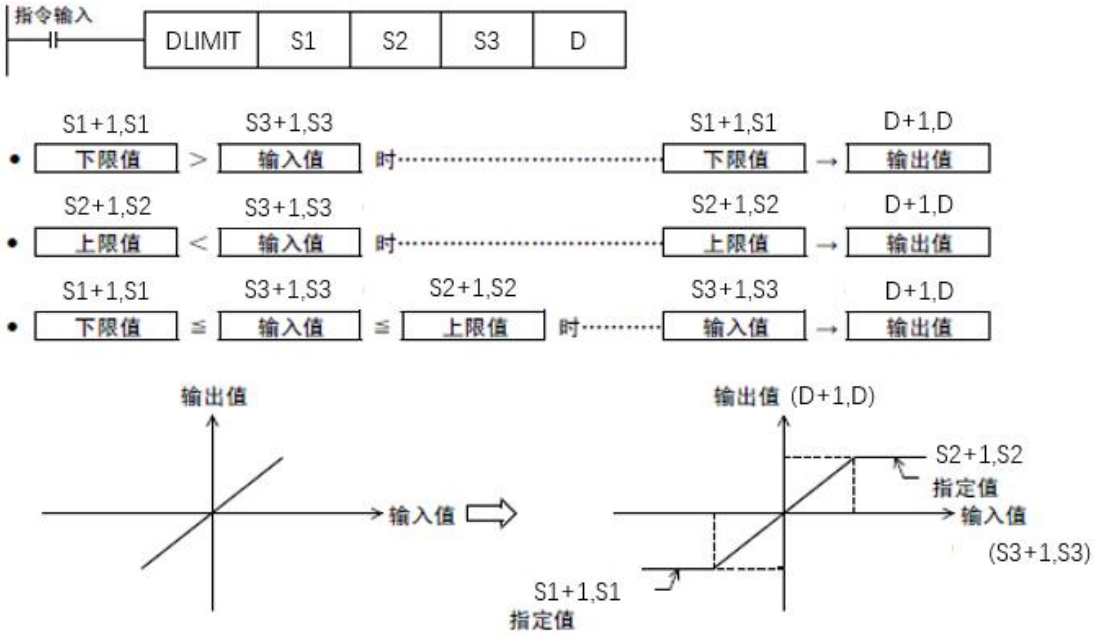
（1）16 位指令

通过判断 S3 中指定的输入值(BIN16 位值)，是否在 S1、S2 指定的上下限值的范围内，以此控制保存在 D 指定的软元件中的输出值。



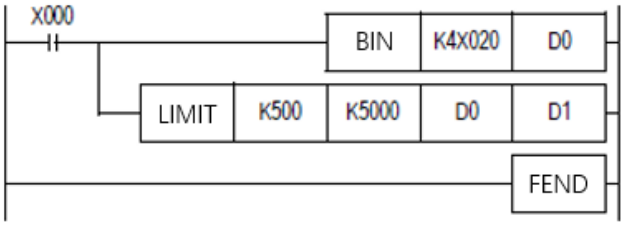
（2）32 位指令

通过判断（S3+1，S3）中指定的输入值(BIN32 位值)，是否在（S1+1，S1）、（S2+1，S2）指定的上下限值的范围内，以此控制保存在（D+1，D)指定的软元件中的输出值。



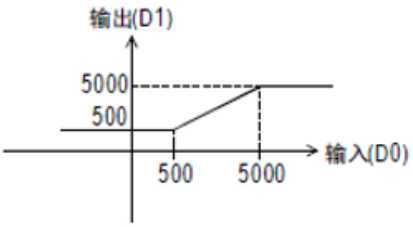
3、程序举例

当X000为ON时，对X020～X037中以BCD值设定的数据执行500～5000的限位值控制，并保存到D1中的程序。

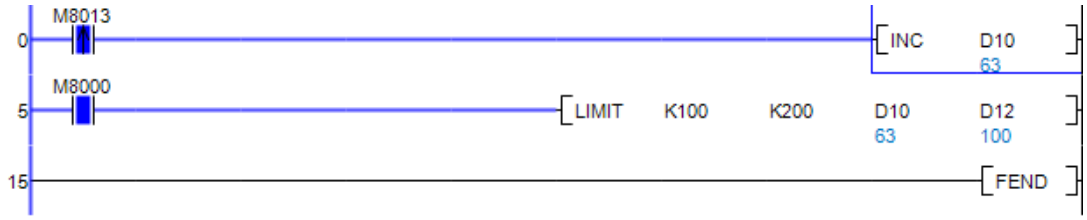


动作

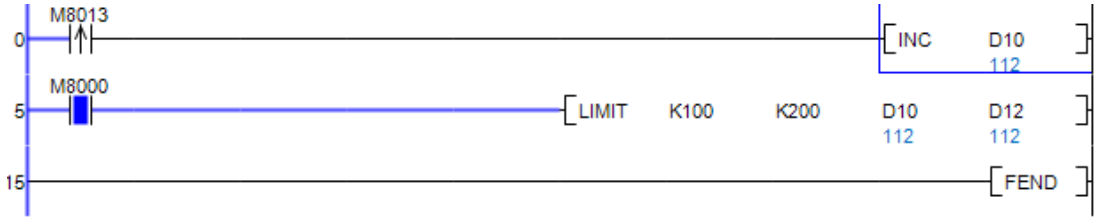
- D0 ＜ 500时，D1为500。
- 500 ≦ D0 ≦ 5000时，D1为D0的值。
- 5000 ＜ D0时，D1为5000。



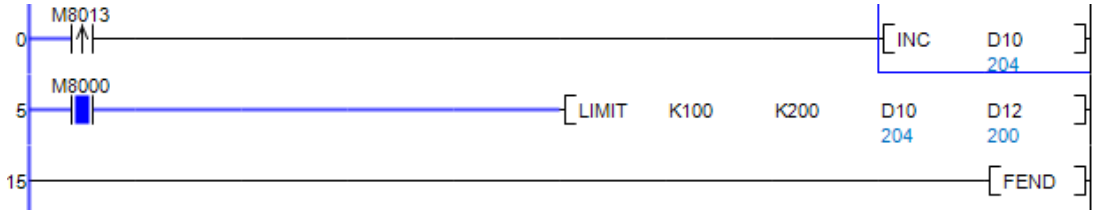
当 S3<S1 时



当 S1<S3<S2 时



当 S3>S2 时



4、注意事项

执行指令后，若设定的下限值 > 上限值时 D8067 报错 6706。

3.20.2 BAND 指令(死区控制)

通过判断输入值是否在指定的死区的上下限范围内，从而来控制输出值的指令。

1、指令格式

16bit 9 步		32bit 17 步		指令格式
BAND	BANDP	DBAND	DBANDP	BAND S1 S2 S3 D

操作数的数据类型如下表

操作数	内容	类型
S1	死区(无输出区域)的下限值	BIN16/32 位

S2	死区(无输出区域)的上限值	BIN16/32 位
S3	要通过死区控制的输入值	BIN16/32 位
D	保存经过死区控制的输出值的软元件编号	BIN16/32 位

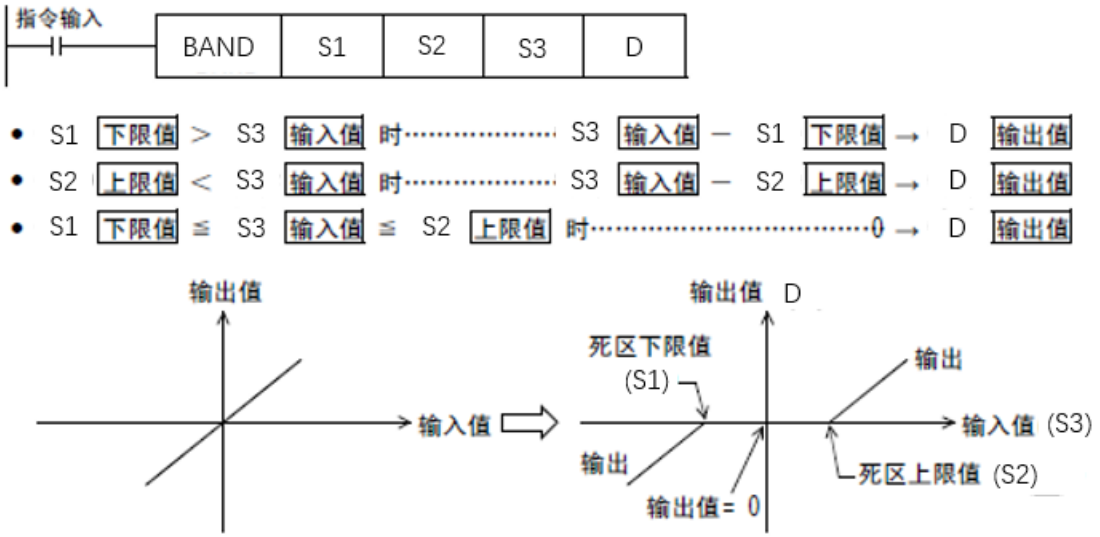
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
S3														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●			●	
S2	●	●	●	●	●	●	●	●	●	●			●	
S3	●	●	●	●	●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明

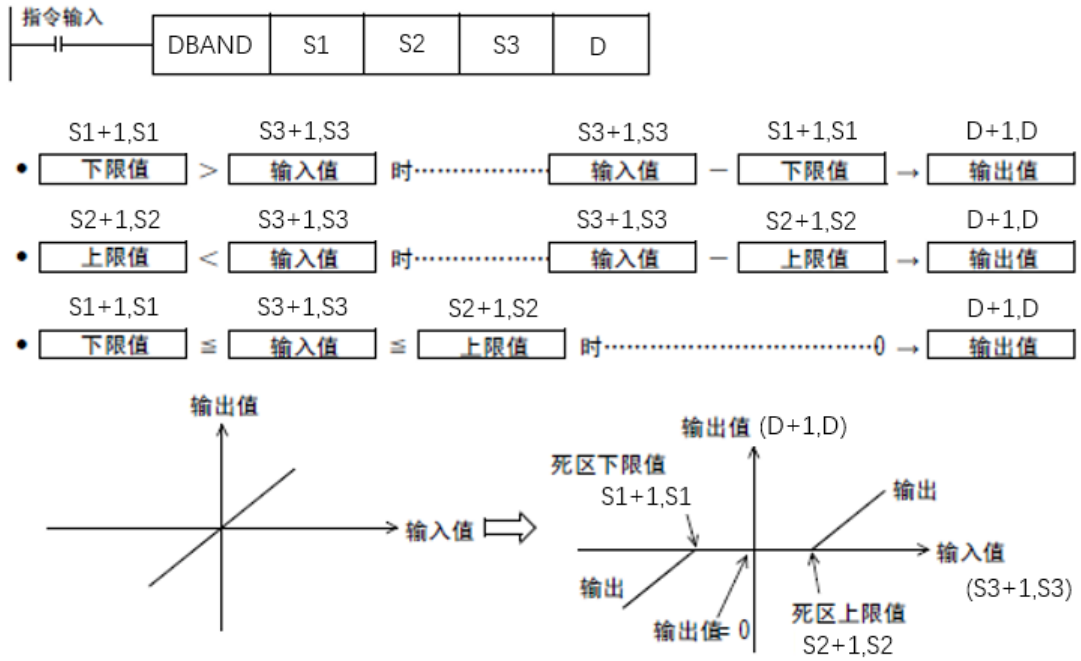
（1）16 位指令

通过判断 S3 指定的输入值(BIN16 位值)是否在 S1、S2 指定的死区范围内，以此来控制保存在 D 指定的软元件中的输出值。



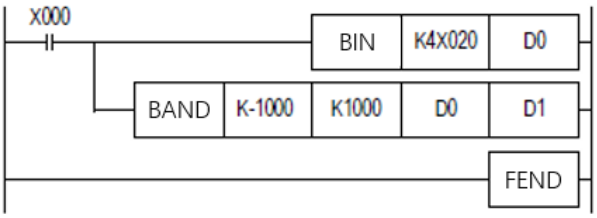
（2）32 位指令

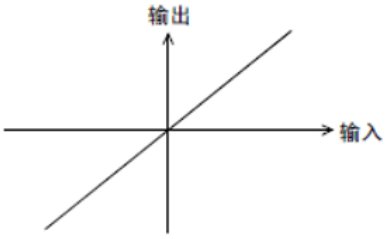
通过判断（S3+1，S3）指定的输入值(BIN32 位值)是否在（S1+1，S1）、（S2+1，S2） 指定的死区范围内，以此来控制保存在（D+1,D)指定的软元件中的输出值。



3、程序举例

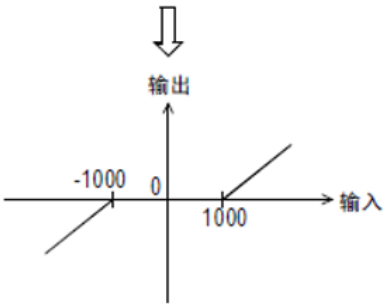
当X000为ON时，对X020～X037中以BCD值设定的数据执行-1000～1000的死区控制，并保存到D1中的程序。



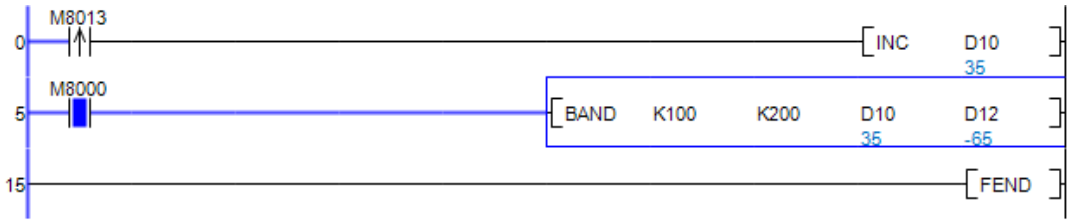


动作

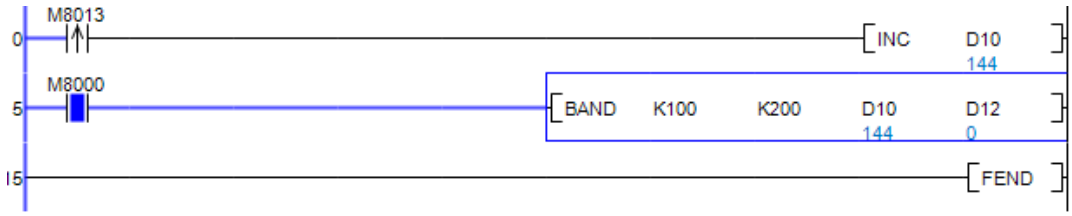
- D0＜(-1000)时，D1中保存(D0)-(-1000)的值。
- -1000≦D0≦1000时，D1中保存0。
- 1000＜D0时，D1中保存(D0)-1000的值。



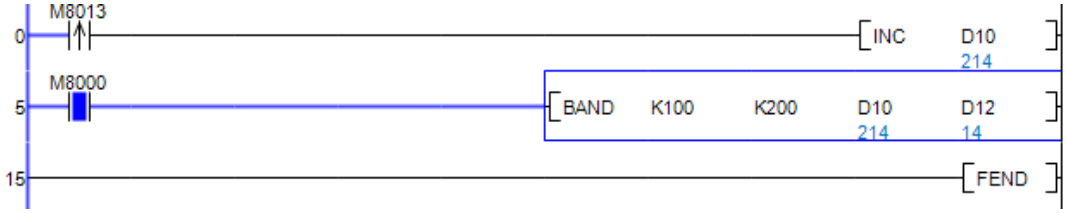
当 S1>S3 时



当 S1<S3<S2 时



当 S2<S3 时



4、注意事项

（1）输出值溢出时，输出值 = 输入值 - 死区下限值

16 位运算：运算结果超出-32768~32767 时

例：S1 = 10，S3 = -32768，输出 D = -32768 - 10 = 8000H - AH = 7FF6H = 32758

32 位运算：运算结果超出-2147483648～2147483647 时

例：（S1+1,S1）= 1000，（S3+1，S3） = -2147483648，输出（D+1，D）= -2147483648 - 1000 = 80000000H - 000003E8H = 7FFFFC18H = 2147482648

（2）指令运行时，如果死区下限值 > 死区上限值时 D8064 报错 6706

3.20.3 ZONE 指令(区域控制)

根据输入值是正数还是负数，用指定的偏差值来控制输出值的指令。

1、指令格式

16bit 9 步		32bit 17 步		指令格式
ZONE	ZONEP	DZONE	DZONEP	ZONE S1 S2 S3 D

操作数的数据类型如下表

操作数	内容	类型
S1	加在输入值上的负偏差值	BIN16/32 位
S2	加在输入值上的正偏差值	BIN16/32 位
S3	要通过区域控制的输入值	BIN16/32 位
D	保存已通过区域控制的输出值的软元件起始编号	BIN16/32 位

操作数的对象软元件如下表

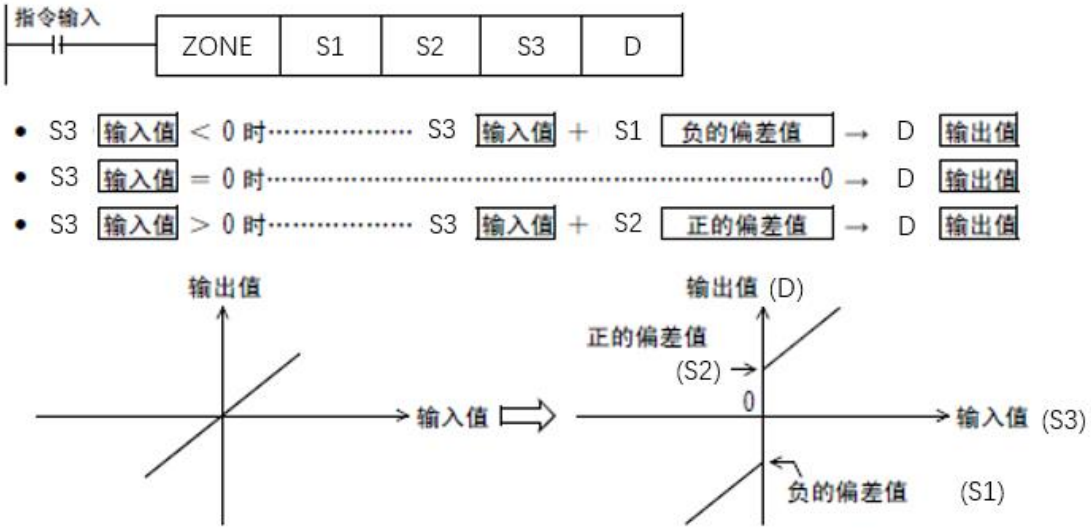
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
S3														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●			

S2	●	●	●	●	●	●	●	●	●	●	●			
S3	●	●	●	●	●	●	●	●	●	●	●			
D		●	●	●	●	●	●	●	●	●	●			

2、功能和动作说明

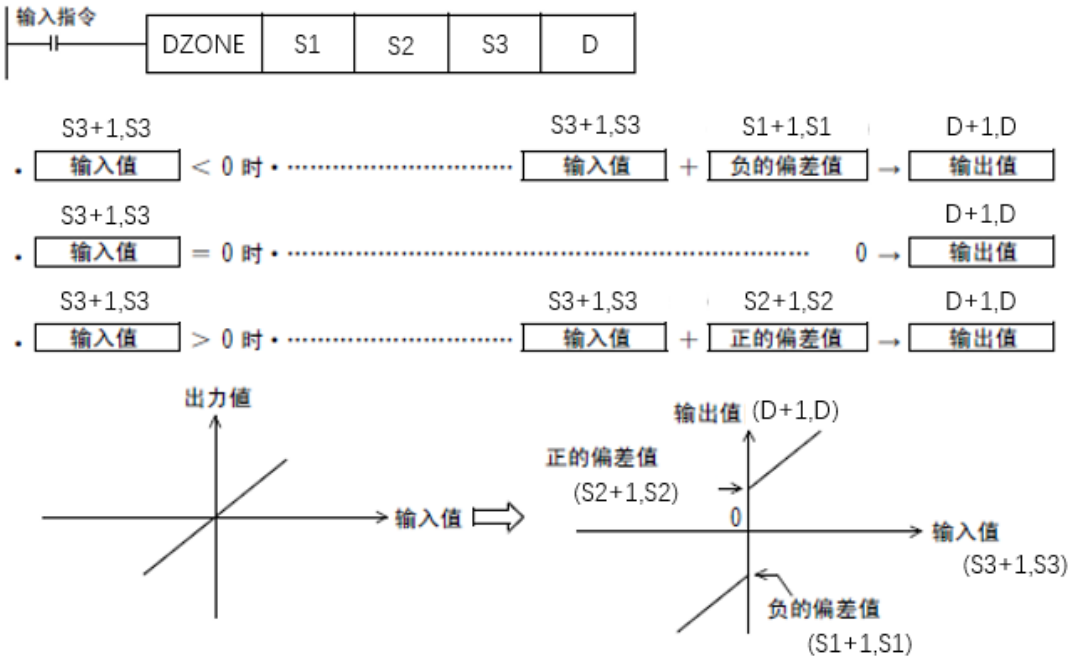
（1）16 位指令

在 S3 指定的输入值上加上 S1 或 S2 指定的偏差值，然后保存到 D 指定的软元件编号中。



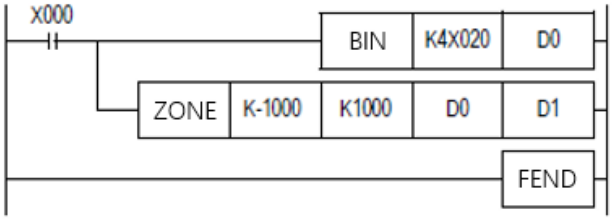
（2）32 位指令

在（S3+1，S3）指定的输入值上加上（S1+1，S1）或（S2+1，S2）指定的偏差值，然后保存到（D+1,D)指定的软元件编号中。



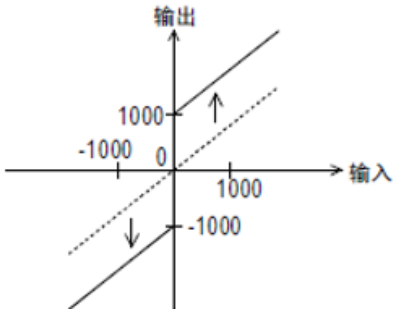
3、程序举例

当X000为ON时，对X020～X37中以BCD值设定的数据执行-1000～1000的区域控制，并保存到D1中的程序。

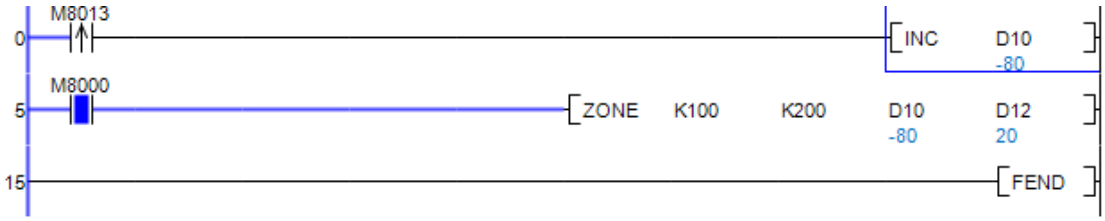


动作

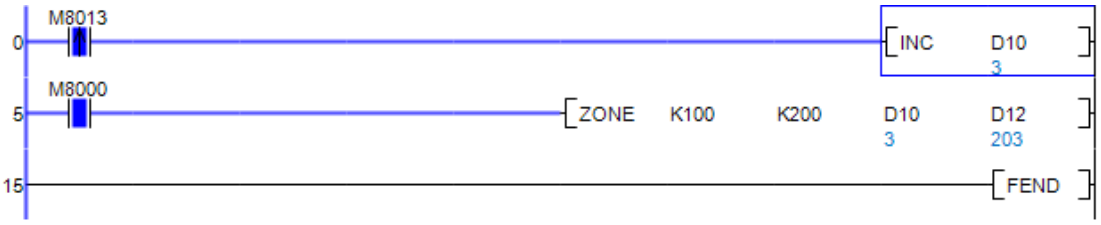
- D0＜0时，D1中保存(D0)+(-1000)的值。
- D0=0时，D1中保存0。
- 0＜D0时，D1中保存(D0)+(1000)的值。



当 S3 为负数时



当 S3 为正数时



4、注意事项

（1）输出值溢出时，输出值 = 输入值 + 负的偏差值

16 位运算：运算结果超出-32768~32767 时

例：S1 = -100，S3 = -32768，输出 D = -32768 + （-100） = 8000H + FF9CH = 7F9CH = 32668

32 位运算：运算结果超出-2147483648～2147483647 时

例：（S1+1,S1）=-1000，（S3+1，S3） =-2147483648，输出（D+1，D）=-2147483648 +（-1000） = 80000000H - FFFFFC18H = 7FFFFC18H = 2147482648

（2）指令运行时，如果死负的偏差值 > 正的偏差值时报错 6706

3.20.4 SCL 指令(定坐标)

根据指定的数据表格，对输入值执行定坐标后输出的指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
SCL	SCLP	DSCL	DSCLP	SCL S1 S2 D

操作数的数据类型如下表

操作数	内容	类型

S1	执行定坐标的输入值或是保存输入值的软元件编号	BIN16/32 位
S2	定坐标用的转换表格软元件的起始编号 BIN16/32 位	BIN16/32 位
D	保存被定坐标控制的输出值的软元件编号	BIN16/32 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2							●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明

（1）16 位指令

根据指定的转换特性，对 S1 指定的输入值执行定坐标，然后保存到 D 指定的软元件编号中。定坐标用的转换，是依据保存在 S2 指定的软元件开始的数据表格执行的。但是，输出数据不是整数值时，小数第 1 位四舍五入后输出。

（2）32 位指令

根据指定的转换特性，对（S1+1，S1）指定的输入值执行定坐标，然后保存到（D+1，D)指定的软元件编号中。定坐标用的转换，是依据保存在（S2+1,S2）指定的软元件开始的数据表格执行的。但是，输出数据不是整数值时，小数第 1 位四舍五入后输出。

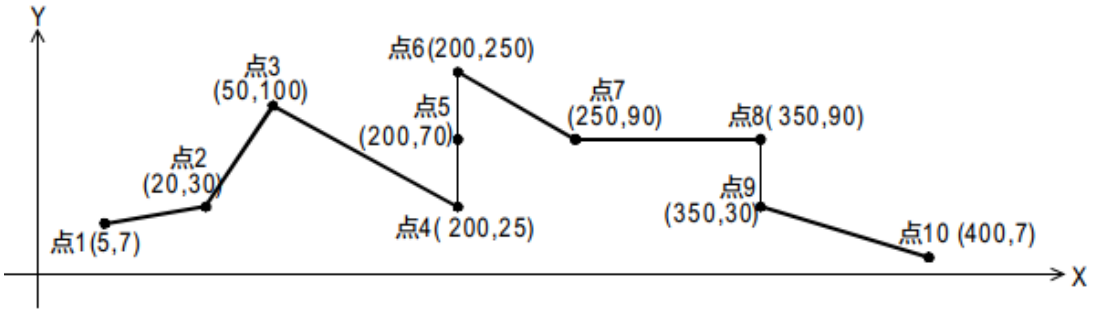
3、程序举例

（1）定坐标用转换表格的设定

设定项目	设定数据表格的软元件分配	
	16 位运算	32 位运算
坐标点数	S2	（S2+1，S2）

点 1	X 坐标	S2+1	（S2+3，S2+2）
	Y 坐标	S2+2	（S2+5，S2+4）
点 2	X 坐标	S2+3	（S2+7，S2+6）
	Y 坐标	S2+4	（S2+9，S2+8）
…	…	…	…
点 n	X 坐标	S2+2n-1	（S2+4n-1，S2+4n-2）
	Y 坐标	S2+2n	（S2+4n+1，S2+4n）

（2）以 16 位运算为例，执行 32 位运算时按 32 位数据格式设定表格数据

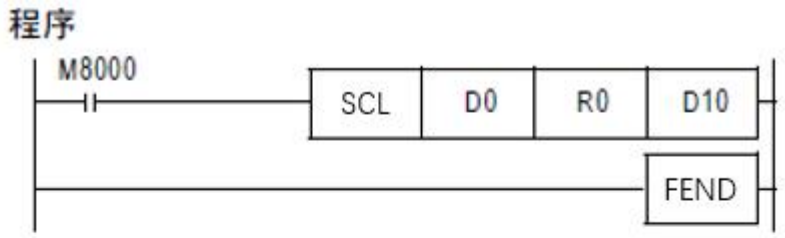


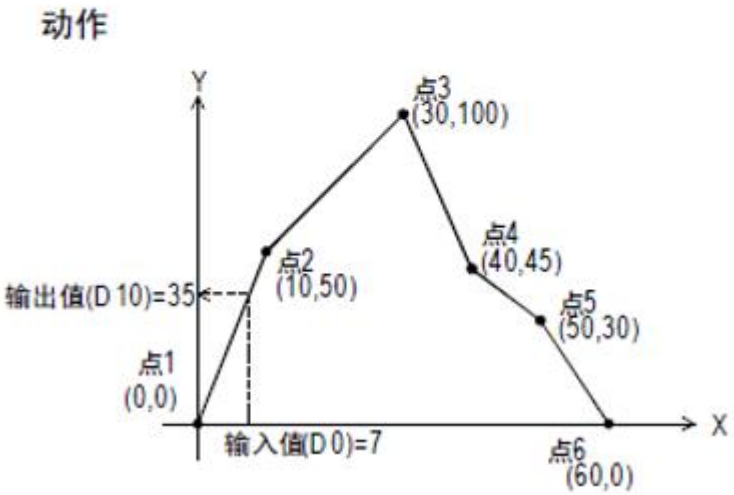
设定项目		设定软元件及设定内容			备注
		S2 中指定了 R0 时		设定内容	
坐标点数		S2	R0	K10	
点 1	X 坐标	S2+1	R1	K5	
	Y 坐标	S2+2	R2	K7	
点 2	X 坐标	S2+3	R3	K20	
	Y 坐标	S2+4	R4	K30	
点 3	X 坐标	S2+5	R5	K50	
	Y 坐标	S2+6	R6	K100	
点 4	X 坐标	S2+7	R7	K200	如果像这样指定 3 点的坐标，则输出值为中间值。 这个例子中，将点 5 的 y 坐标指
	Y 坐标	S2+8	R8	K25	
点 5	X 坐标	S2+9	R9	K200	

	Y 坐标	S2+10	R10	K70	定为输出值(中间值)。 此外，3 点以上的 X 坐标相同时，也输出第 2 点的数值
点 6	X 坐标	S2+11	R11	K200	
	Y 坐标	S2+12	R12	K250	
点 7	X 坐标	S2+13	R13	K250	如果像这样指定 2 点的坐标，则输出值取后一个点的 y 坐标值。 这个例子中，将点 9 的 y 坐标指定为输出值
	Y 坐标	S2+14	R14	K90	
点 8	X 坐标	S2+15	R15	K350	
	Y 坐标	S2+16	R16	K90	
点 9	X 坐标	S2+17	R17	K350	
	Y 坐标	S2+18	R18	K30	
点 10	X 坐标	S2+19	R19	K400	
	Y 坐标	S2+20	R20	K7	

3、程序举例

根据 R0 开始的软元件设定的定坐标用转换表格，对 D0 输入的值执行定坐标，然后输出到 D10 中的程序。





设定项		软元件	设定内容
坐标点数		R0	K6
点 1	X 坐标	R1	K0
	Y 坐标	R2	K0
点 2	X 坐标	R3	K10
	Y 坐标	R4	K50
点 3	X 坐标	R5	K30
	Y 坐标	R6	K100
点 4	X 坐标	R7	K40
	Y 坐标	R8	K45
点 5	X 坐标	R9	K50
	Y 坐标	R10	K30
点 6	X 坐标	R11	K60
	Y 坐标	R12	K0

4、注意事项

（1）当 X 坐标或 Y 坐标相同时，按上图表格中所写方式输出结果。

（2）以下情况报错 6706：

• 数据表格的 Xn 数据没有按照升序排列时。但是，由于运算是从数据表格的软元件编号的低位侧开始检索的，所以即使数据表格的一部分没有按照升序排列，但到这个部分为止的运算不会出现运算错误，指令会被执行。
• S1 在数据表格设定的范围以外时。
• 运算过程中的数值超出了 32 位数据的范围时。此时，请确认各点之间的距离没有超出 65535 以上。如果超出 65535 时，请缩短各点之间的距离。

3.20.5 DABIN 指令 10 进制 ASCII 到 BIN 的转换）

将以 10 进制数字的 ASCII 码(30H～39H)形式现实的数据转换成 BIN 数据的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
DABIN	DABINP	DDABIN	DDABINP	DABIN S D

操作数的数据类型如下表

操作数	内容	类型
S	保存要转换成 BIN 值的数据(ASCII 码)的软元件起始编号	字符串
D	保存转换结果的软元件编号	BIN16/32 位

操作数的对象软元件如下表

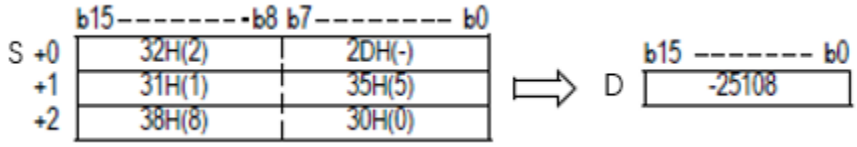
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S					●	●	●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●	●	●	●	

2、功能和动作说明

（1）16 位指令

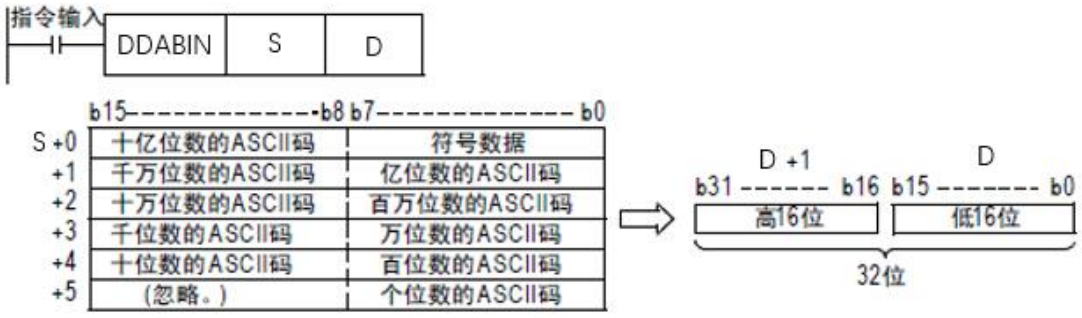
将以 10 进制数字的 ASCII 码 (30H ～ 39H) 显示的 D~D+2 的数据转换成 16 位数据 (BIN) 后，保存到 D 中。

例如， S～ S+2 为-25108 的 ASCII 码时，在中如下所示地保存 16 位数据(BIN)。

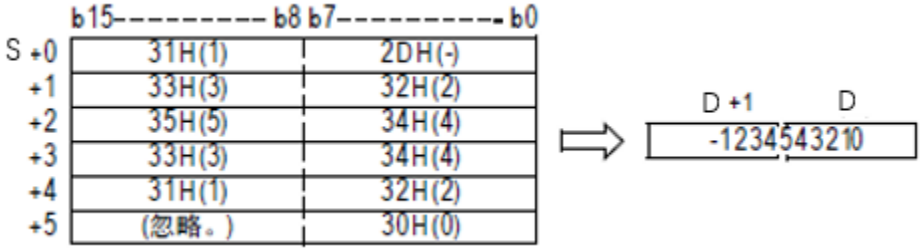


（2）32 位指令

将以 10 进制数字的 ASCII 码（30H ～ 39H）显示的D~D+5的数据转换成32位数据（BIN）后，保存到（D+1，D）中。

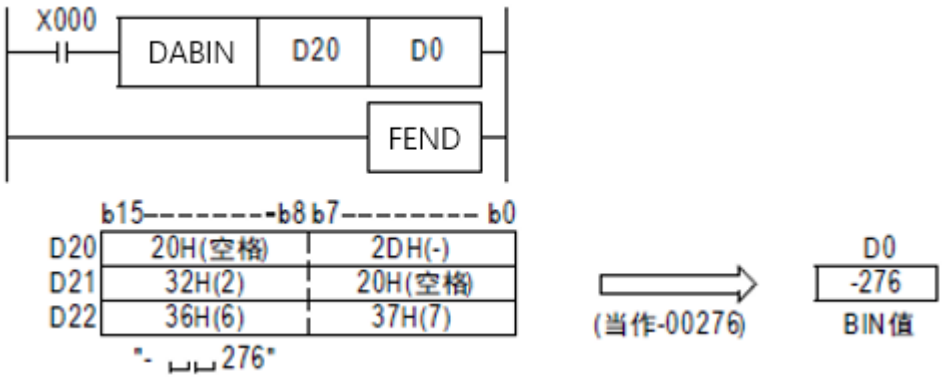


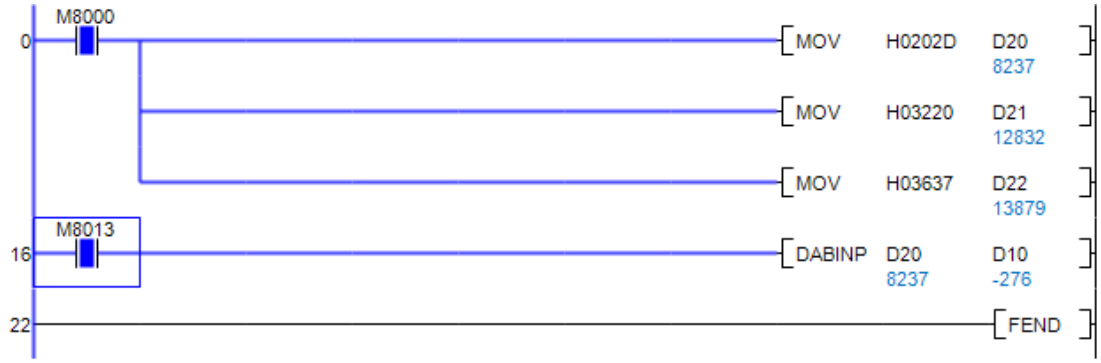
例如， S～ S+5为-1,234,543,210的ASCII码时，在[D+1、D]中如下所示地保存32位数据(BIN)。



3、程序举例

当X000为ON时，将D20～D22中设定的符号以及5位10进制数字的ASCII码数据转换成BIN值后，保存到D0中的程序。





4、注意事项

（1）各位数的 ASCII 码位“20H”、“00H（NULL）”时，作为“30H”处理。

（2）以下情况报错 6706：

• 符号数据(S 的低位字节)中，要转换的数据为正时设定“20H(空格)”，为负时设定"2DH(-)"，设定其它以外值报错。

• S～S+2(5)的各位数的 ASCII 码为“30H”～“39H”、“20H(空格)”“00H(NULL)”以外的值时。

• S～S+2(5)的数值范围为下述的范围以外时：

16 位运算：-32768~32767；32 位运算：-2147483648~2147483647。

• S～S+2(5)超出软元件范围时。

3.20.6 BINDA 指令(BIN 到 10 进制 ASCII 转换)

将 BIN 数据转换成 ASCII 码(30H～39H)的指令。

1、指令格式

16bit 5 步		32bit 9 步		指令格式
DBINDA	DABINP	DBINDA	DBINDAP	BINDA S D

操作数的数据类型如下表

操作数	内容	类型
S	保存要转换成 ASCII 码的 BIN 数据的软元件编号	BIN16/32 位
D	保存转换结果的软元件编号	字符串

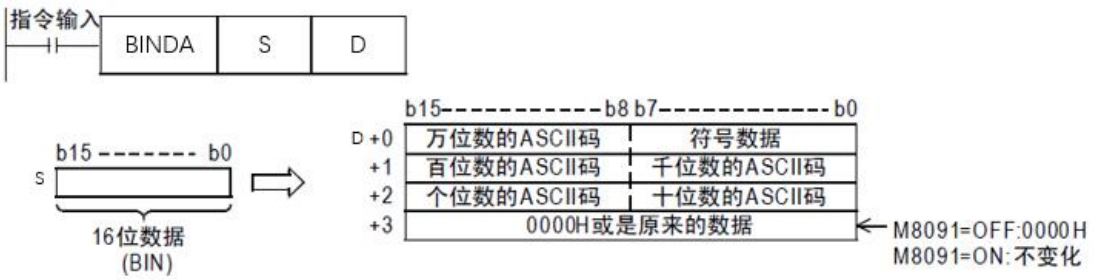
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S	●	●	●	●	●	●	●	●	●	●	●	●	●	
D					●	●	●	●	●	●			●	

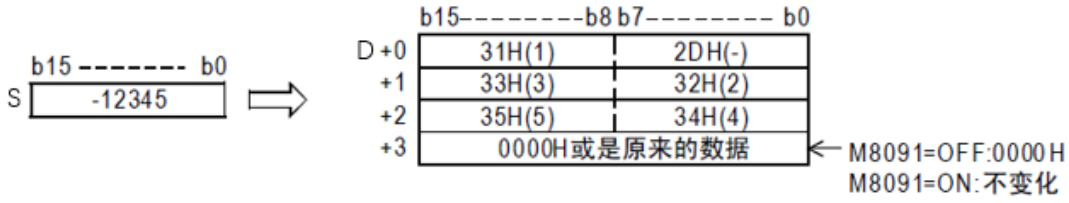
2、功能和动作说明

（1）16 位指令

将 S 的 16 位数据(BIN)按照 10 进制的各个位数转换成 ASCII 码(30H～39H)，然后保存到 D 中。

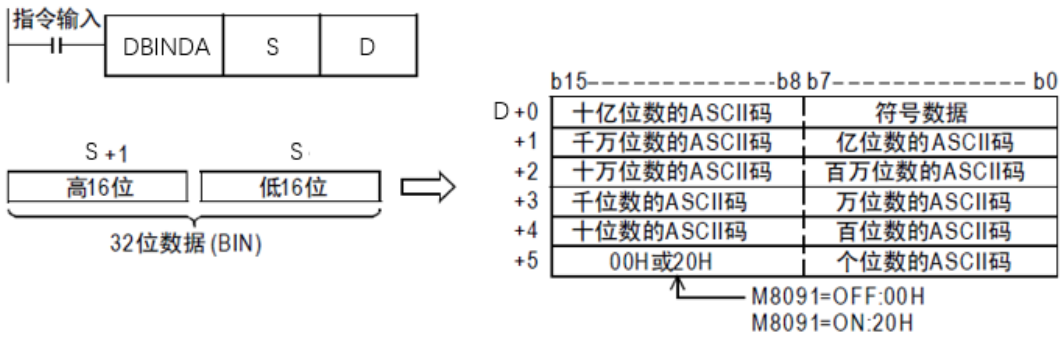


例如，S 为-12345 时，在 D 以后如下所示的进行保存。

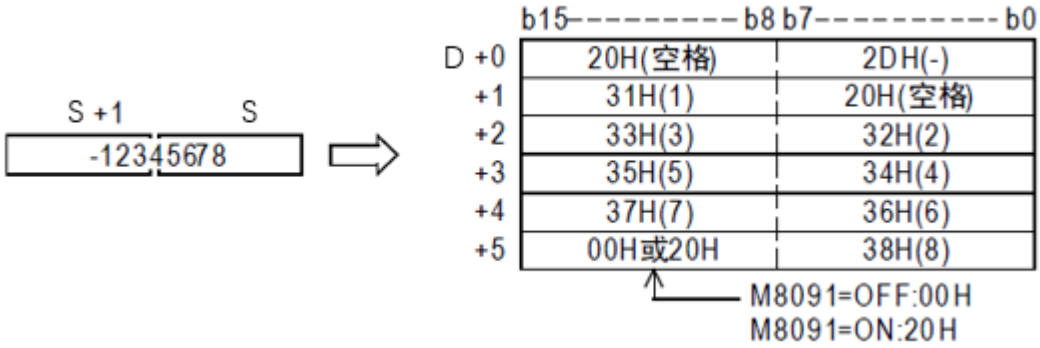


（2）32 位指令

将（S+1，S）的 32 位数据(BIN)按照 10 进制的各个位数转换成 ASCII 码(30H～39H)，然后保存到 D 开始的软元件中。

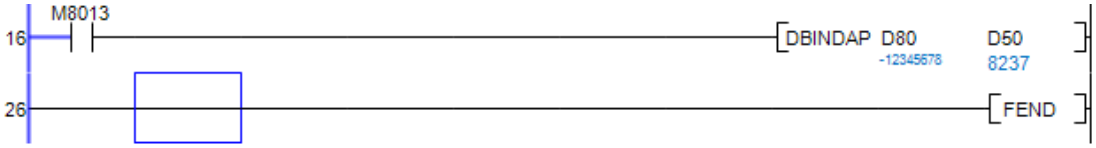


例如，[S+1、S]为-12,345,678 时， D 开始的软元件中如下所示地进行保存。



3、程序举例

[S+1、S]为-12,345,678 时， D 开始的软元件中如下所示地进行保存。



元件	当前值	数据类型	显示方
D50	202D	16位整数	HEX
D51	3120	16位整数	HEX
D52	3332	16位整数	HEX
D53	3534	16位整数	HEX
D54	3736	16位整数	HEX
D55	0038	16位整数	HEX

4、注意事项

（1）相关软元件的动作：

软元件	名称	内容	
		16 位运算	32 位运算
M8091	输出字符数切换信号	M8091=OFF 时,D+3 为 0000H（NULL）。 M8091=ON 时，D+3 不变化。	M8091=OFF 时，D+5 高字节为 00H（NULL）。 M8091=ON 时，D+5 高字节为 20H（空格）。

要注意 16 位运算时 M8091 为 OFF 的情况，要占用 4 个字软元件。

（2）符号数据(D 的低位字节)中，要转换的数据为正时保存“20H(空格)”,为负时保存"2DH(-)"。

（3）当 S 中数据的总位数低于 D 能存储的总位数时，有效数的左侧保存“20H（空格）”，如上面的例子。

（4）D 的占用点数不能超出改软元件的范围，超出报错 6706。

3.20.7 SCL2 指令(定坐标 2)

根据指定的数据表格，对输入值执行定坐标后输出的指令。

1、指令格式

16bit 7 步		32bit 13 步		指令格式
SCL2	SCL2P	DSCL2	DSCL2P	SCL2 S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	执行定坐标的输入值或是保存输入值的软元件编号	BIN16/32 位
S2	定坐标用的转换表格软元件的起始编号	BIN16/32 位
D	保存被定坐标控制的输出值的软元件编号	BIN16/32 位

操作数的对象软元件如下表

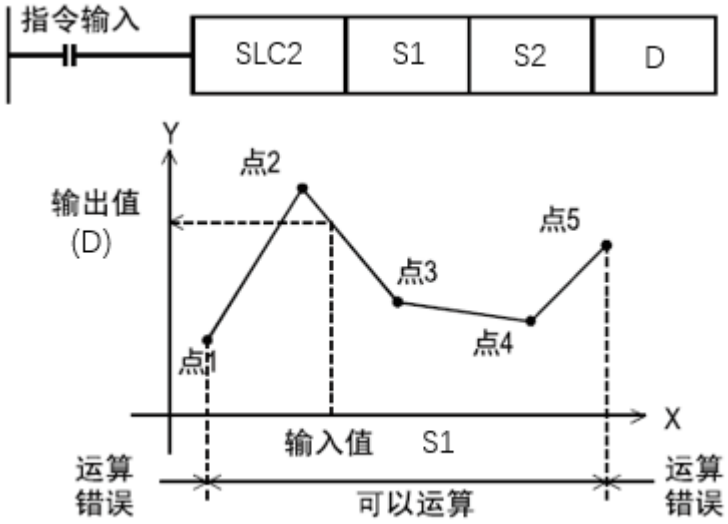
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		

S2													
D													
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●			●	
S2							●	●	●	●			●	
D		●	●	●	●	●	●	●	●	●			●	

2、功能和动作说明

（1）16 位指令

根据指定的转换特性，对 S1 指定的输入值执行定坐标，然后保存到 D 指定的软元件编号中。定坐标用的转换，是依据保存在 S2 指定的软元件开始的数据表格执行的。但是，输出数据不是整数值时，小数第 1 位四舍五入后输出。

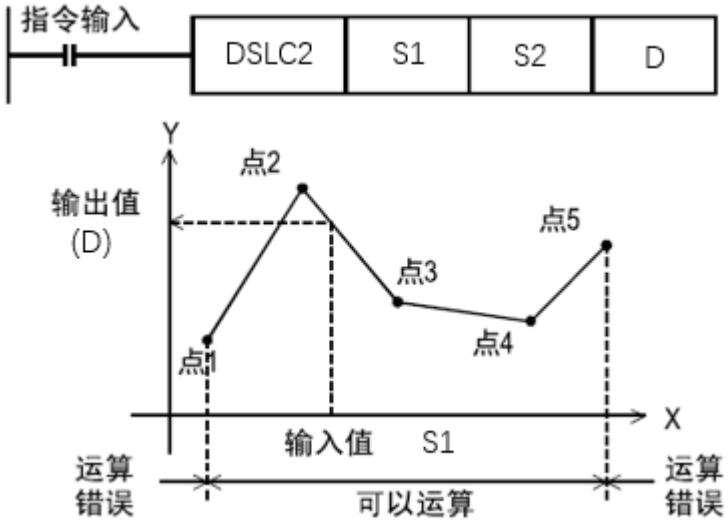


设定项		设定数据表格的软元件分配
坐标点数		S2
X 坐标	点 1	S2+1
	点 2	S2+2
	点 3	S2+3
	点 4	S2+4
	点 5	S2+5
Y 坐标	点 1	S2+6
	点 2	S2+7

	点 3	S2+8
	点 4	S2+9
	点 5	S2+10

（2）32 位指令

根据指定的转换特性，对（S1+1，S1）指定的输入值执行定坐标，然后保存到（D+1，D)指定的软元件编号中。定坐标用的转换，是依据保存在（S2+1,S2）指定的软元件开始的数据表格执行的。但是，输出数据不是整数值时，小数第 1 位四舍五入后输出。



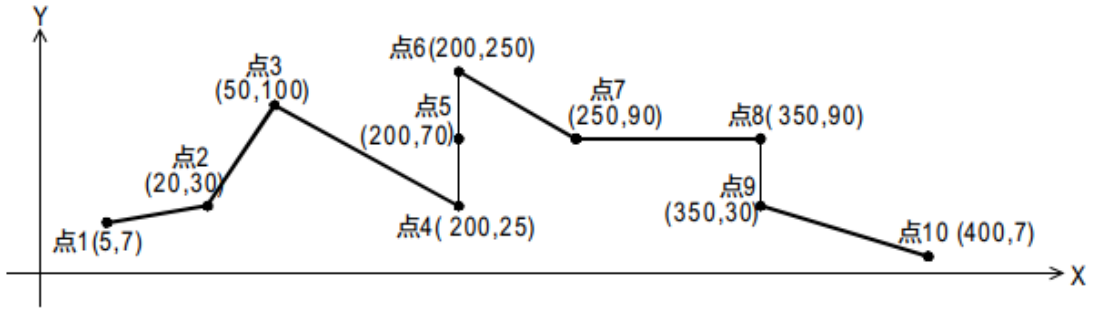
设定项		设定数据表格的软元件分配
坐标点数		S2+1, S2
X 坐标	点 1	S2+3, S2+2
	点 2	S2+5, S2+4
	点 3	S2+7, S2+6
	点 4	S2+9, S2+8
	点 5	S2+11, S2+10
Y 坐标	点 1	S2+13, S2+12
	点 2	S2+15, S2+14
	点 3	S2+17, S2+16
	点 4	S2+19, S2+18
	点 5	S2+21, S2+20

（1）定坐标用转换表格的设定

设定项目	设定数据表格的软元件分配

坐标点数		S2	（S2+1，S2）
X 坐标	点 1	S2+1	（S2+3，S2+2）
	点 2	S2+2	（S2+5，S2+4）
	···	···	···
	点 n（最后）	S2+n	（S2+2n+1，S2+2n）
Y 坐标	点 1	S2+n+1	（S2+2n+3，S2+2n+2）
	点 2	S2+n+2	（S2+2n+5，S2+2n+4）
	···	···	···
	点 n（最后）	S2+2n	（S2+4n+1，S2+4n）

（2）以 16 位运算为例，执行 32 位运算时按 32 位数据格式设定表格数据



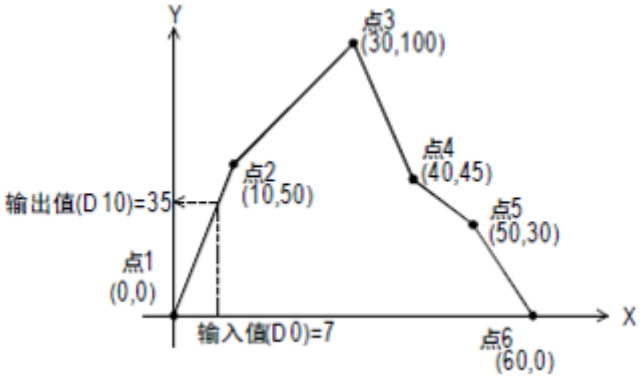
设定项目		设定软元件及设定内容			备注
		S2 中指定了 R0 时		设定内容	
坐标点数		S2	R0	K10	
X 坐标	点 1	S2+1	R1	K5	
	点 2	S2+2	R2	K20	
	点 3	S2+3	R3	K50	
	点 4	S2+4	R4	K200	参考*1
	点 5	S2+5	R5	K200	
	点 6	S2+6	R6	K200	
	点 7	S2+7	R7	K250	
	点 8	S2+8	R8	K350	参考*2

	点 9	S2+9	R9	K350	
	点 10	S2+10	R10	K400	
Y 坐标	点 1	S2+11	R11	K7	
	点 2	S2+12	R12	K30	
	点 3	S2+13	R13	K100	
	点 4	S2+14	R14	K25	参考*1
	点 5	S2+15	R15	K70	
	点 6	S2+16	R16	K250	
	点 7	S2+17	R17	K90	
	点 8	S2+18	R18	K90	参考*2
	点 9	S2+19	R19	K30	
	点 10	S2+20	R20	K7	

*1. 如果像点 4、5、6 那样指定 3 点的坐标时则输出值为中间值。在这个例子中，将点 5 的 y 坐标指定为输出值(中间值)。此外，即使 3 点以上的 X 坐标相同时，输出第 2 点的数值。
*2. 如果像点 8、9 那样指定 2 点的坐标，则输出值取后一个点的 y 坐标的值。在这个例子中，将点 9 的 y 坐标指定为输出值。

3、程序举例

动作



设定项		软元件	设定内容
坐标点数		R0	K6
X 坐标	点 1	R1	K0
	点 2	R2	K10
	点 3	R3	K30
	点 4	R4	K40
	点 5	R5	K50
	点 6	R6	K60
Y 坐标	点 1	R7	K0
	点 2	R8	K50
	点 3	R9	K100
	点 4	R10	K45
	点 5	R11	K30
	点 6	R12	K0

4、注意事项

（1）当 X 坐标或 Y 坐标相同时，按例程中所写方式输出结果。

（2）以下情况报错 6706：

• 数据表格的 Xn 数据没有按照升序排列时。但是，由于运算是从数据表格的软元件编号的低位侧开始检索的，所以即使数据表格的一部分没有按照升序排列，但到这个部分为止的运算不会出现运算错误，指令会被执行。

• S1 在数据表格设定的范围以外时。
• 运算过程中的数值超出了 32 位数据的范围时。此时，请确认各点之间的距离没有超出 65535 以上。如果超出 65535 时，请缩短各点之间的距离。

3.21 扩展文件寄存器指令

ER 寄存器存储在外部 Flash 对于一些想长期保存的数据可以使用 ER 寄存器保存。

段编号与起始软元件编号对照表：

段编号	起始软元件编号	写入软元件范围	段编号	起始软元件编号	写入软元件范围
段 0	R0	ER0~ER2047	段 8	R16384	ER16384~ER18431
段 1	R2048	ER2048~ER4095	段 9	R18432	ER18432~ER420479
段 2	R4096	ER4096~ER6143	段 10	R20480	ER20480~ER22527
段 3	R6144	ER6144~ER8191	段 11	R22528	ER22528~ER24575
段 4	R8192	ER8192~ER10239	段 12	R24576	ER24576~ER26623
段 5	R10240	ER10240~ER12287	段 13	R26624	ER26624~ER28671
段 6	R12288	ER12288~ER14335	段 14	R28672	ER28672~ER30719
段 7	R14336	ER14336~ER16383	段 15	R30720	ER30720~ER32767

3.21.1 LOADR 指令(读出扩展文件寄存器)

将扩展文件寄存器 ER 的当前值读到 R 寄存器中。

1、指令格式

16bit 5 步		32bit		指令格式
LOADR	LOADRP	\	\	LOADR S n

操作数的数据类型如下表

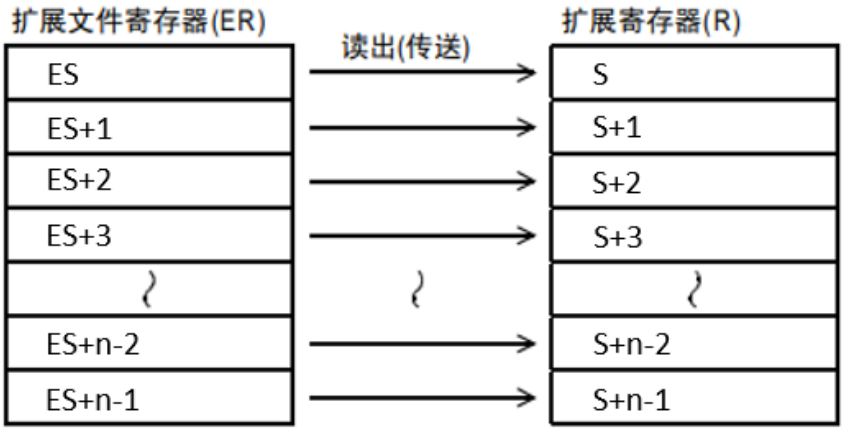
操作数	内容	类型
S	保存数据的扩展寄存器的软元件编号(作为数据传送源的扩展文件寄存器与 S 的编号相同)	BIN16 位

n	读出(传送)点数（0≦n≦32767)	BIN16 位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
n											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S								●					●	
n							●							

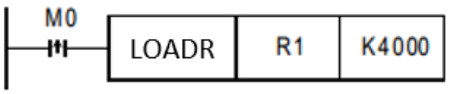
2、功能和动作说明



以点为单位进行读出（传送），最多可以读出（传送）32768 点，当 n=0 时，按 n=32768 执行指令。

3、程序举例

将 ER1~ER4000 的内容读到 R1~R4000



4、注意事项

建议使用脉冲型指令，以免连续型指令执行次数太多，减少 Flash 的寿命。

3.21.2 SAVER 指令(成批写入扩展文件寄存器)

将任意点数的扩展寄存器(R)的当前值 1 段(2048 点)写入扩展文件寄存器(ER)中时使用。

1、指令格式

16bit 7 步		32bit		指令格式
SAVER	SAVERP	\	\	SAVER S n D

操作数的数据类型如下表

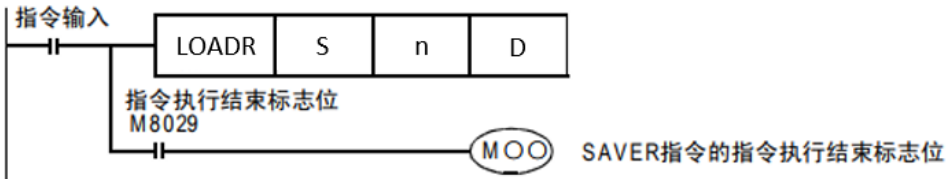
操作数	内容	类型
S	保存数据的扩展寄存器的软元件编号，但是，只可以指定扩展寄存器的段的起始软元件编号	BIN16 位
n	每个运算周期的写入(传送)点数 0≦n≦2048	BIN16 位
D	保存已经写入的点数的软元件编号	BIN16 位

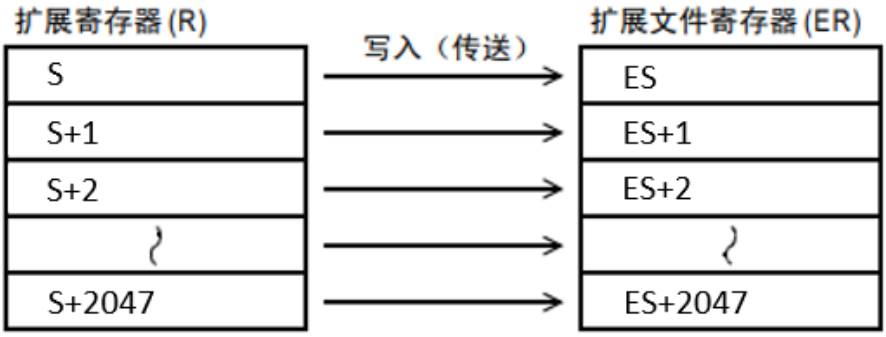
操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S														
n											●	●		
D														
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S								●					●	
n														
D							●						●	

2、功能和动作说明

通过 2048/n 个运算周期(如有余数时，为+1 个扫描)将 S～S+2047 的扩展寄存器（R）的内容(当前值)写入具有相同编号的扩展文件寄存器(ER)中。指令执行过程中，在 D 中保存已经写入的点数。

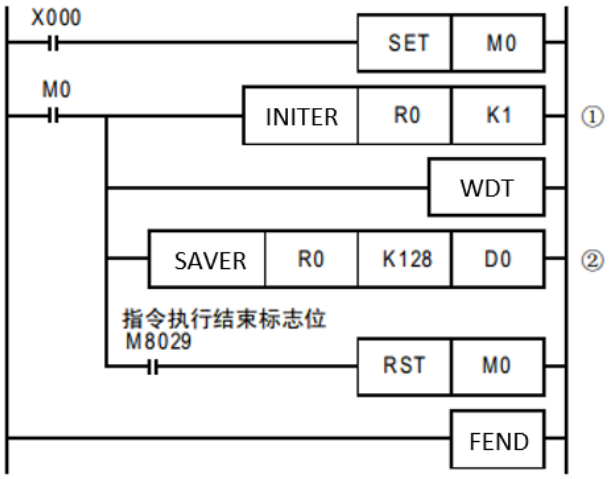




- 每个运算周期写入 n 点。
- 以段为单位(2048 点)执行写入。各段的起始软元件编号对应 3.22 扩展文件寄存器指令章节的表格。
- 在 n 中指定了 0 的值时，作为 n=2048 执行指令。
- 当 2048 点的写入(传送)结束时，指令执行结束，指令执行结束标志位 M8029 变为 ON。

3、程序举例

当 X000 为 ON 时，将设定数据用的扩展寄存器 R10～R19(0 段)的变更内容，反映到扩展文件寄存器(ER)中的程序。(每个运算周期写入 128 点)



①先执行 INITER 指令初始化扩展文件寄存器（ER）

②使用 SAVER 指令每个周期写入 128 点，将 R0~R2047 的值都会写入 ER0~ER2047。

4、注意事项

（1）SAVER 指令执行之前必须先执行 INITER 指令或者 INITR 指令进行 ER 数据的初始化，否则报错 6770。

（2）为防止 SAVER 指令执行写入时间过长造成看门狗超时，在此之前刷新看门狗时间或更改比较大的时间保证能完成写入。

4 子程序

4.1 概述

动作说明：当指令输入为 ON 时，执行 CALL 指令，向标记 P 的步跳转。接着，执行标记 P 的子程序，执行 SRET 后，返回到 CALL 指令的下一步，子程序的调用返回程序可以参考 3.2.2 CALL 指令(子程序调用)

注意事项：

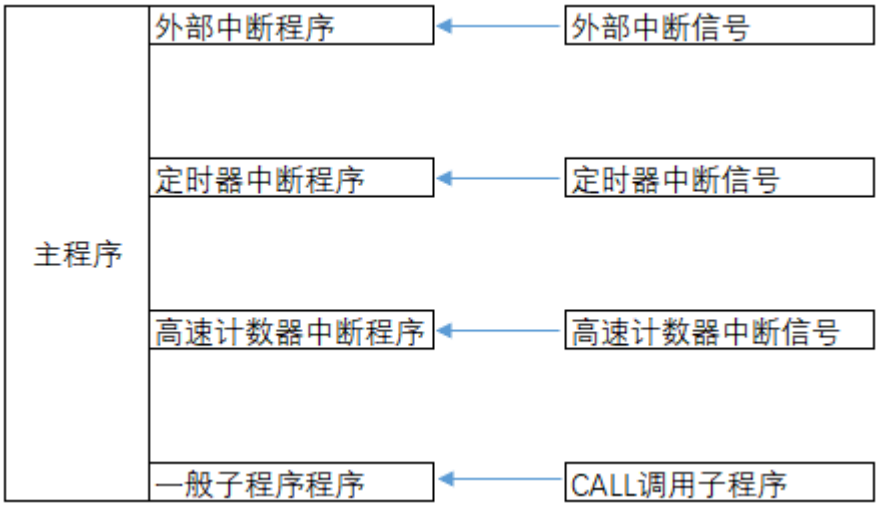
（1）CALL 指令用的标记 P；

（2）最多允许 6 层嵌套，超出报 6634 停机错误；

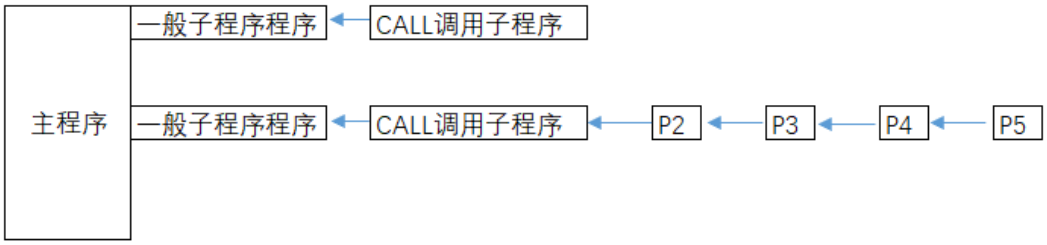
关于流程类的梯形图 mPLC 在自检的时候就会把错误找出来并停机错误。

概述			
P	CALL 指令用	最多支持 4096 个子程序；其中子程序属性可以被设置为普通子程序、加密子程序。 加密子程序与普通子程序容量一样不受限制,共同占用系统 64K 步的容量。	
P	CJ 指令用	4096 点,配合子程序使用	
I	中断子程序	外部中断	X000-X007 输入中断，编号 I00 □， I10 □，I20 □，I30 □，I40□，I50 □，6 点，（□表示：0 下降沿中断，1 上升沿中断）。 沿中断禁止标志位寄存器置 ON 后，则对应的输入 中断被禁止。
		定时中断	I6□□，I7□□，I8□□，3 点(□□=1~99，时基=1ms)
		计数完成中断	I010、I020、I030、I040、I050。5 点（DHSCS 指令用）

主程序与主程序关系



嵌套子程序示例，最高可嵌套六层

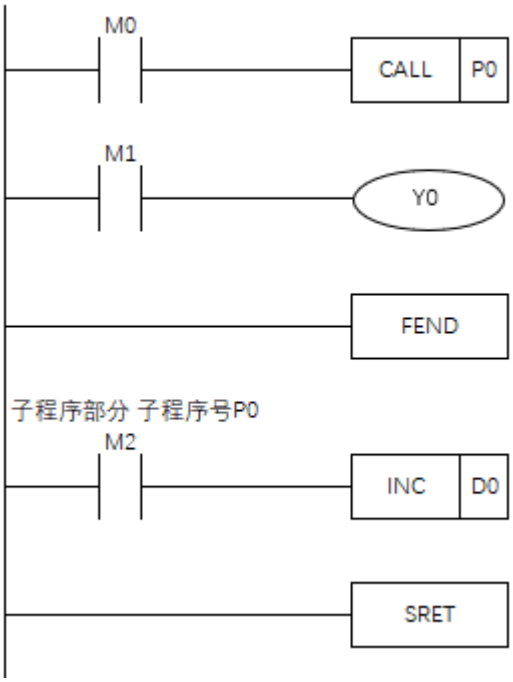


子程序使用，写程序时注意格式：必须在FEND指令后对标记编程。Pn作为一段子程序的开始，以SRET作为一段子程序的结束。用CALL Pn调用子程序。其中n可以为0~9999中的任一值。

使用子程序调用，可以简化编程，可以将几个地方需要用的公共部分写在子程序中，再调用子程序即可实现。

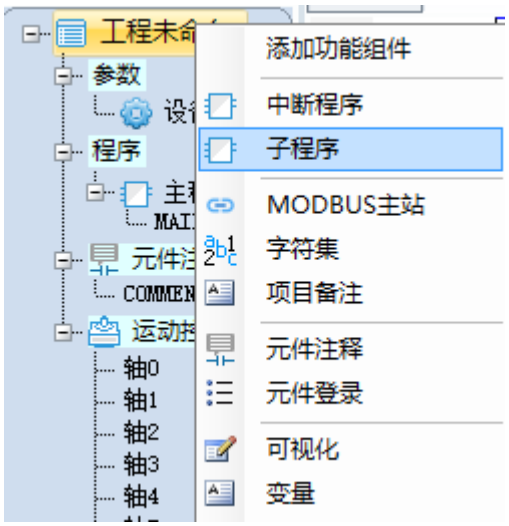
子程序编程举例以及用法

- 当M0为ON，则执行调用指令，跳转到标记为P1的子程序步。在这里，执行子程序后，通过执行SRET指令后，返回到原来的主程序步，接着继续执行后续的主程序。
- 子程序可用于6层嵌套调用。
- 调用子程序时，主程序所属的OUT、定时器等均保持。
- 子程序返回时，子程序所属的OUT、定时器等均保持。
- 子程序中不要写脉冲、计数、定时等一个扫描周期内无法完成的指令。

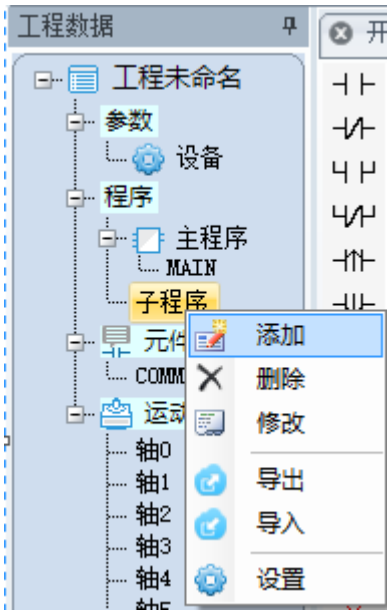


4.2 通用子程序

在左侧栏的工程数据中找到工程名，点击右键，将子程序功能块添加至工程内。

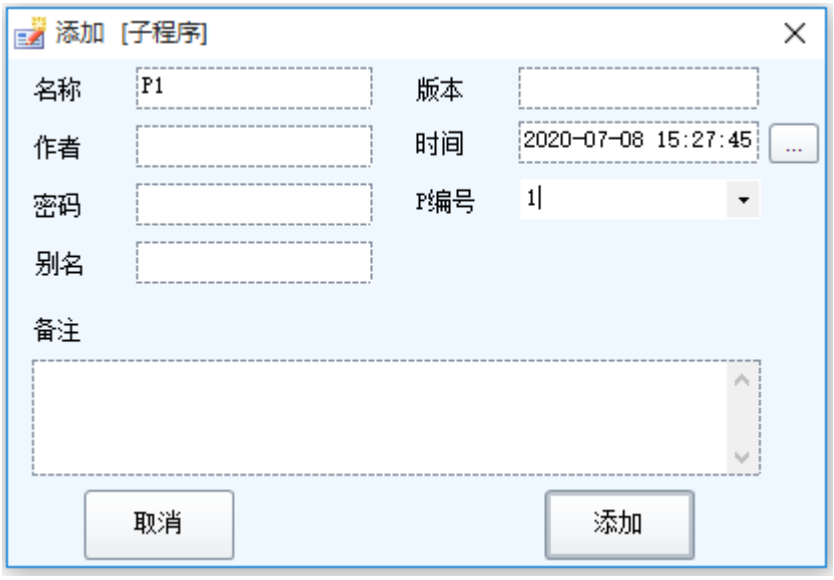


在子程序选项中选择添加。

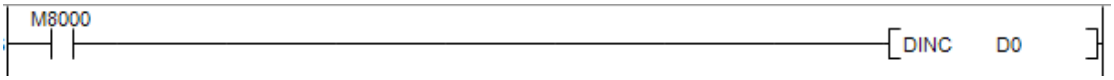


输入子程序的名称以及 P 的编号即可。

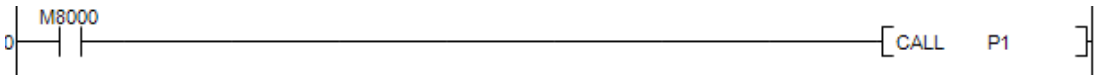
P 编号范围 0~4095。



子程序的内容

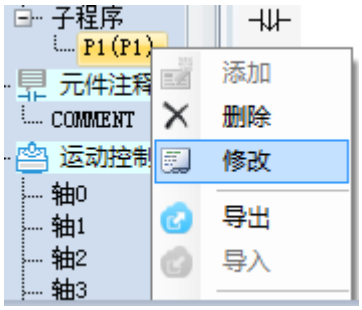


主程序的内容(调用子程序 P1)



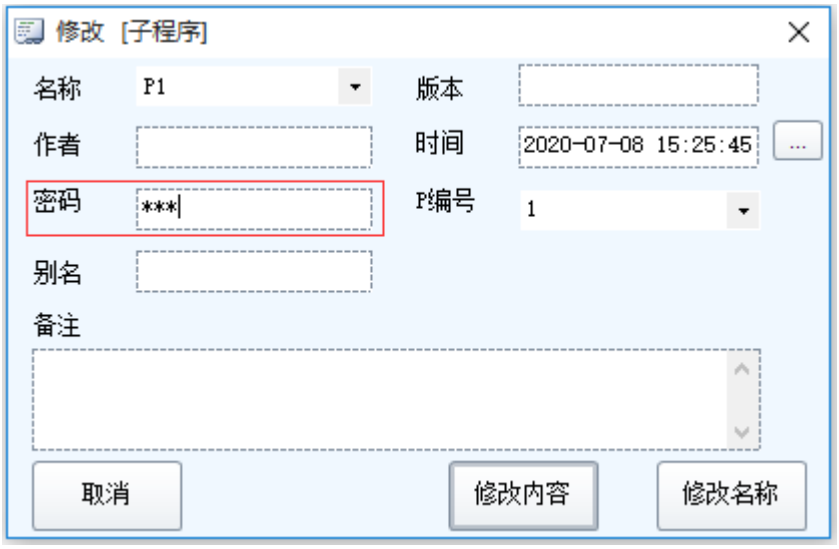
4.3 加密子程序

在要修改的子程序点击右键，选择修改。



在密码一栏填入需要设置的密码，点击修改内容，密码即修改成功。

加密后的子程序使用起来与未加密的使用方式一致。



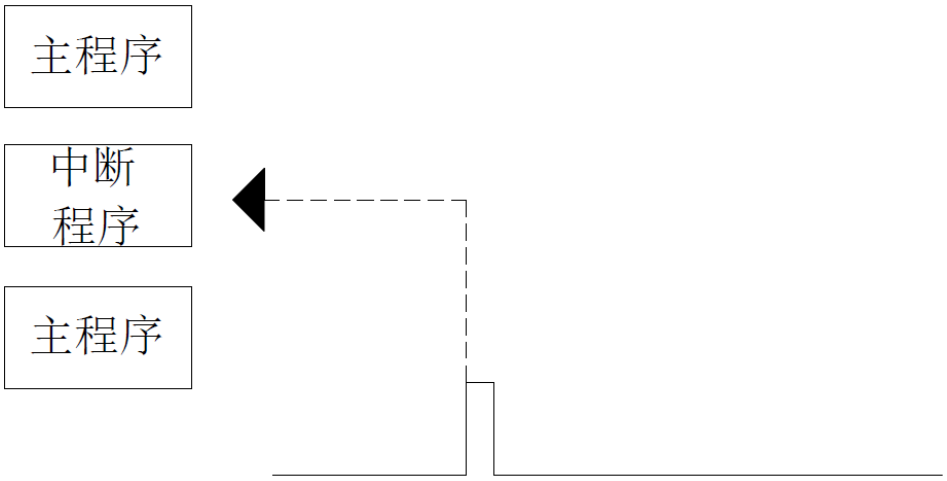
4.4 中断子程序

详见 6 中断中断程序。

5 中断

动作说明：EI 是中断开放指令，DI 是中断禁止指令。中断源触发中断后，PLC 就跳转到其中断用指针编号指定的子程序执行。IRET 是中断返回指令，它是中断处理程序的结束指令，其作用是使 PLC 返回到被中断时的下一条指令继续执行。 若没有中断处理子程序，虽然触发了中断，但没有处理任何事情。

输入端子 X 可以作为外部中断的输入用，每一输入端对应于一个外部中断，输入的上升沿或者下降沿都可触发中断，中断子程序写在中断程序下。当产生中断后，主程序立即停止执行，转而执行相应的中断子程序，等中断子程序执行完成后，再立即返回继续执行主程序。



多个中断依次发生时，以先发生的为优先。完全同时发生时，优先级别高的为优先。优先级从高到低分别为高速计数器中断、外部中断、时间中断、脉冲输出完成中断。

在中断例行程序的执行过程中，禁止其它的中断。

中断的种类

1） 外部信号输入中断：可定义X0~X5输入信号的上升沿或下降沿进行中断，对于不需要即时响应的X信号，还可以采用脉冲捕捉的功能。

2） 定时器中断：以1ms～99ms的固定周期发生的中断。

3） 高速计数中断：与DHSCS比较置位指令并用，当高速计数的当前值达到设定值时，产生中断。

5.1 中断事件分类：

5.1.1 外部输入中断(6 点)

在不受可编程控制器运算周期的影响下，接收来自特定的输入编号的输入信号。触发该输入信号，执行中断子程序。由于输入中断可以处理比运算周期更短的信号，因此可在顺控过程中作为需要优先处理或者短时间脉冲处理控制时使用。

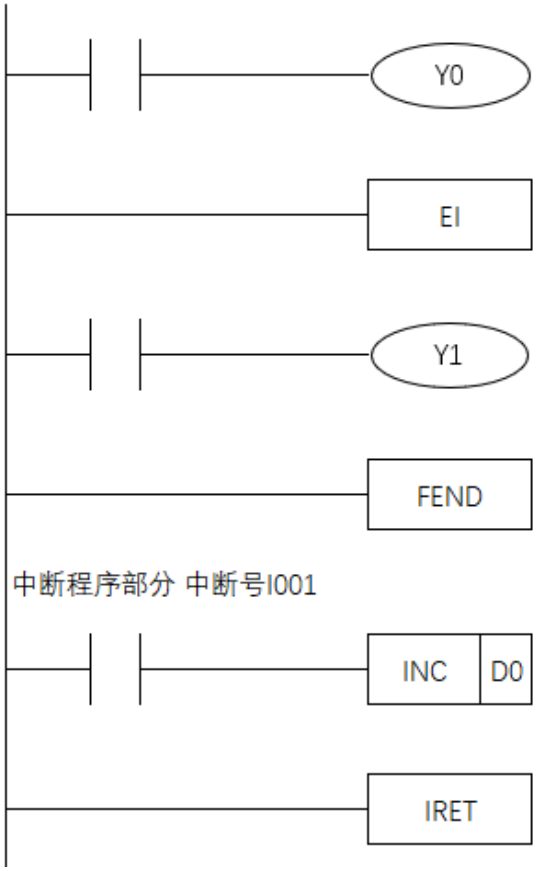
输入	输入中断用指针		禁止中断标志位
	上升沿中断	下降沿中断	
X00	I001	I000	M8050
X01	I101	I100	M8051
X02	I201	I200	M8052
X03	I301	I300	M8053
X04	I401	I400	M8054
X05	I501	I500	M8055

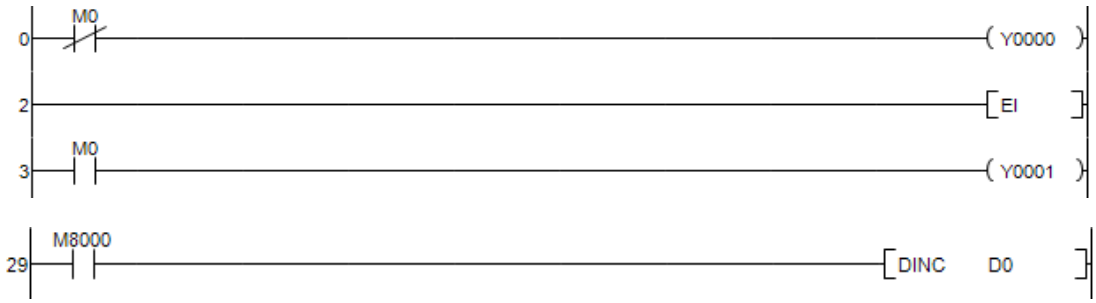
同一个梯形图中同一个输入的上升沿和下降沿都可以触发对应的外部中断。

程序举例:

如果用EI指令允许中断，则在扫描程序的过程中如果中断输入由“OFF→ON”，则执行中断例行程序结束后回到主程序继续执行。

中断用指针（I****），需要在中断程序下新建中断程序对应此中断号。
当主程序在运行时，外部信号X0由“ON→OFF”此时触发中断号I001（X0下降沿中断），执行中断子程序里的内容，当执行完毕后，返回到主程序中执行剩下的程序。





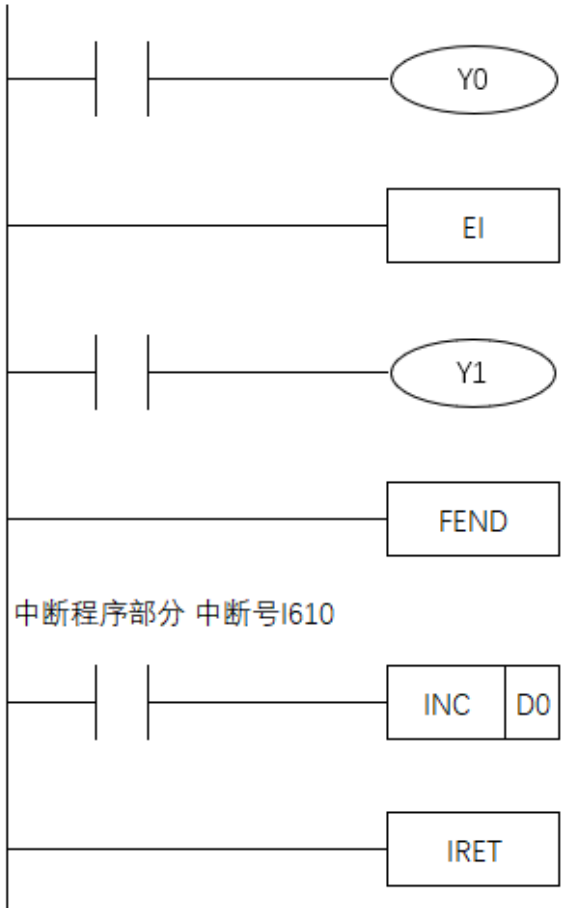
5.1.2 定时器中断(3 点)

每隔指定的中断循环时间(1ms～99ms)，执行中断子程序。在可编程控制器的运算周期以外，需要循环中断处理的控制中使用。

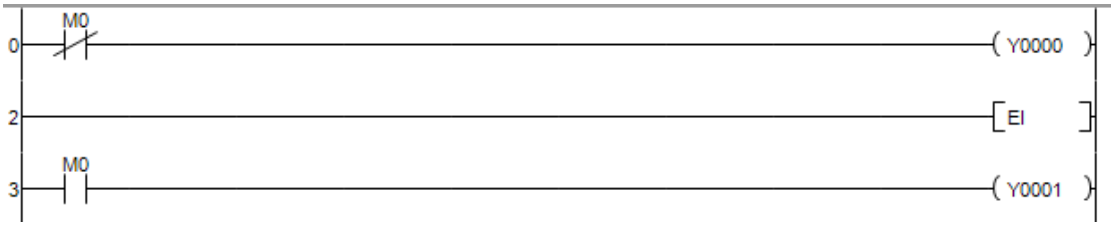
输入编号	中断周期（ms）	中断禁止标志位
I6XX	在指针名的 XX 中，输入 1~99 的整数。 例如:I610=每 10ms 的定时器中断	SM8056
I7XX		SM8057
I8XX		SM8058

程序举例：

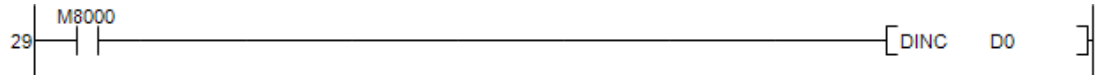
如果用EI指令允许中断，则在扫描程序的过程中每10ms进入一次中断子程序，则执行中断例行程序结束后回到主程序继续执行。
中断用指针（I6XX）、（I7XX）、（I8XX），在XX内填写1~99即为XXms的定时中断程序。
当主程序在运行时，每过10ms触发中断号I610（10ms定时器中断），执行中断子程序里的内容，当执行完毕后，返回到主程序中执行剩下的程序。



主程序



10ms 定时中断程序



5.1.3 计数器中断

详见 6.9 高速计数中断

5.2 中断优先级

外部输入中断 > 定时器中断 > 计数器中断

I000 > I100 > I200 > I300 > I400 > I500 > I001 > I101 > I201 > I301 > I401 > I501 > I6xx > I7xx > I8xx > I010 > I020 > I030 > I040 > I050 > I060

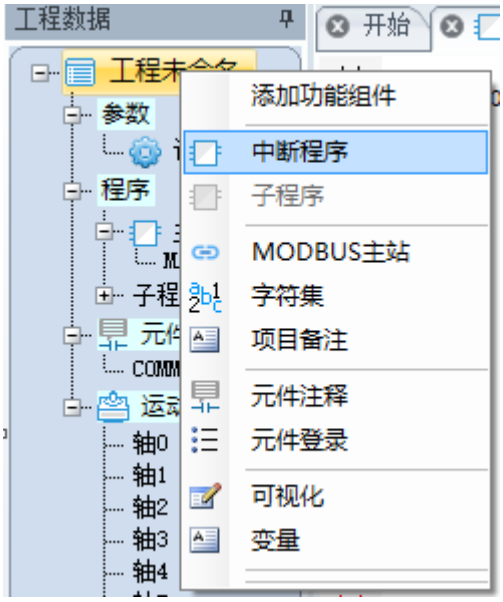
注意事项

（1）如果对应的中断禁止标志置位，即使产生了中断也不会执行。

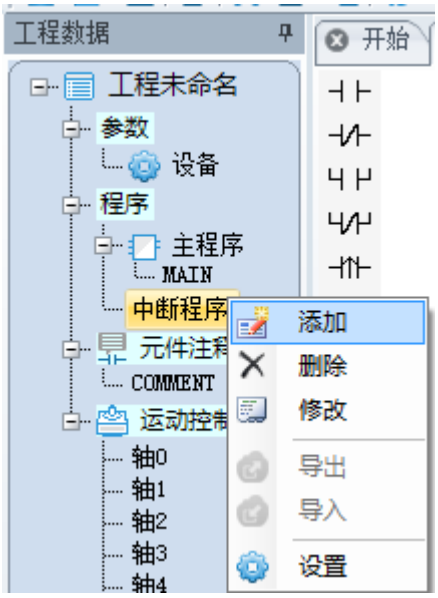
（2）注意 M8393 寄存器的动作，中断程序后面紧接一个 LD M8393，表示中断产生后不立即动作，而是延迟 D8393 的延时时间才动作。

5.3 如何建立一个中断

首先在工程数据下的项目名称右键，找到列表中的中断程序即可添加进工程。



接着找到工程下的程序下的中断程序，点击右键，选择添加。



在这里填写中断程序的名称以及功能（I编号）。
I编号如下表所示。

外部输入点的上升沿以及下降沿对应不同的中断号。

输入上下沿中断		
端口	上升沿	下降沿
X00	I001	I000
X01	I101	I100
X02	I201	I200
X03	I301	I300

X04	I401	I400
X05	I501	I500

定时器中断对应不同的中断号，可设 1~99ms 的定时中断时间。

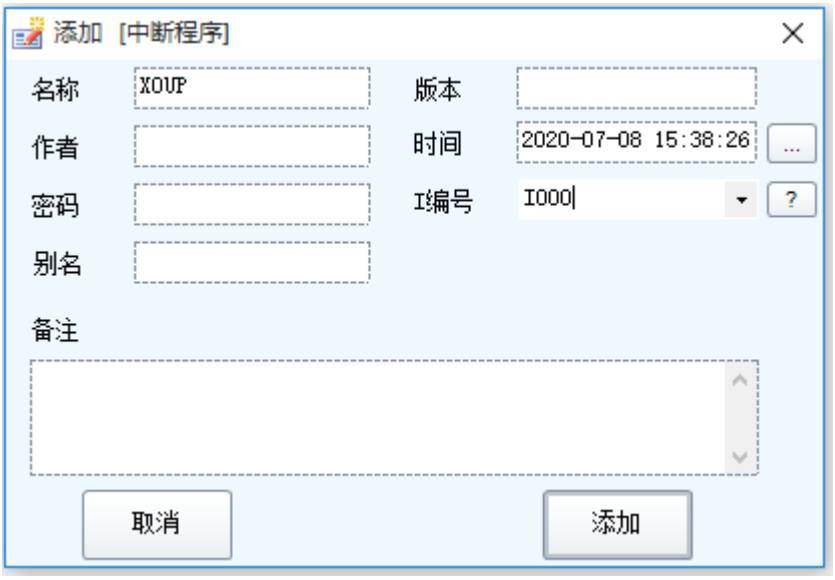
定时器中断
I6xx
I7xx
I8xx

计数器中断配合 DHSCS 指令来使用，指定比较值、计数器号、中断号。

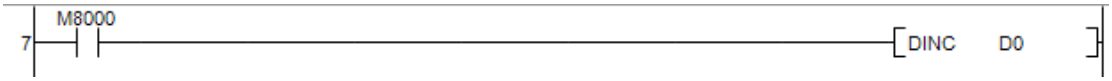
计数器中断
I010
I020
I030
I040
I050

在 I 编号中按照需要的功能填写相对应的中断号。

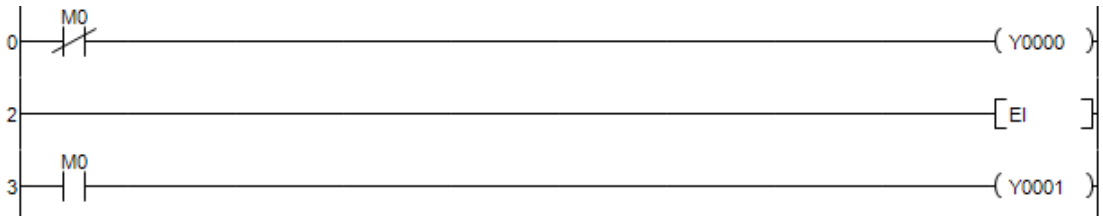
例 X0 中断,输入名称以及中断号即可。（中断号 I000 类型为 X0 上升沿触发）



在新添加的中断程序中写入程序，每次 X0 上升沿时执行中断号为 I000 的中断子程序。



在主程序中打开中断使能 EI。



将程序下载进 PLC 即可。

6 高速计数

6.1 功能概述

MP2 系列 PLC 具多有 24 路高速输入端口 X00-X27，实现最多 8 路单相和 8 路 AB 相计数，支持最高脉冲输入频率 200KHz，支持单相递增计数、单相递减计数、AB 相计数(1 倍频、4 倍频)三种计数模式，以及高速中断功能。

6.2 高速计数输入端口分配

单相模式

	C235	C236	C237	C238	C239	C240	C241	C242
X000	U/D							
X001								
X002								
X003		U/D						
X004								
X005								
X006			U/D					
X007								
X010								
X011				U/D				
X012								
X013								
X014					U/D			
X015								
X016								
X017						U/D		
X020								

X021								
X022							U/D	
X023								
X024								
X025								U/D
X026								
X027								

AB 相模式

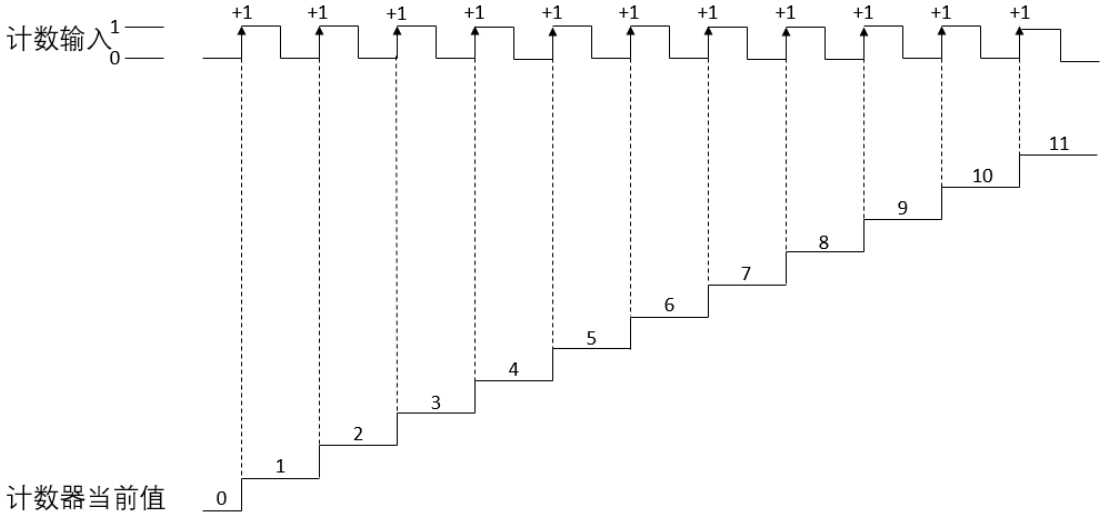
	C235	C236	C237	C238	C239	C240	C241	C242
X000	A							
X001	B							
X002	Z							
X003		A						
X004		B						
X005		Z						
X006			A					
X007			B					
X010			Z					
X011				A				
X012				B				
X013				Z				
X014					A			
X015					B			
X016					Z			
X017						A		
X020						B		
X021						Z		
X022							A	
X023							B	
X024							Z	
X025								A
X026								B
X027								Z

6.3 高速计数模式

MP2 系列 PLC 支持单相递增、单相递减、AB 相（1 倍频、4 倍频)三种计数模式。

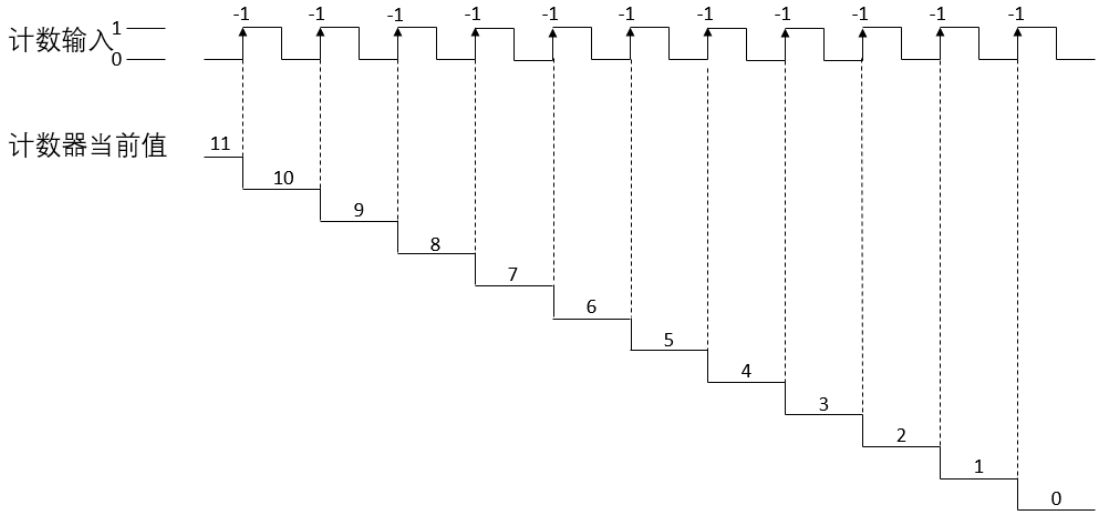
1)单相递增模式

此模式下，计数输入脉冲信号，计数值随着每个脉冲信号的上升沿递增计数。



2)单相递减模式

此模式下，计数输入脉冲信号，计数值随着每个脉冲信号的上升沿递减计数。

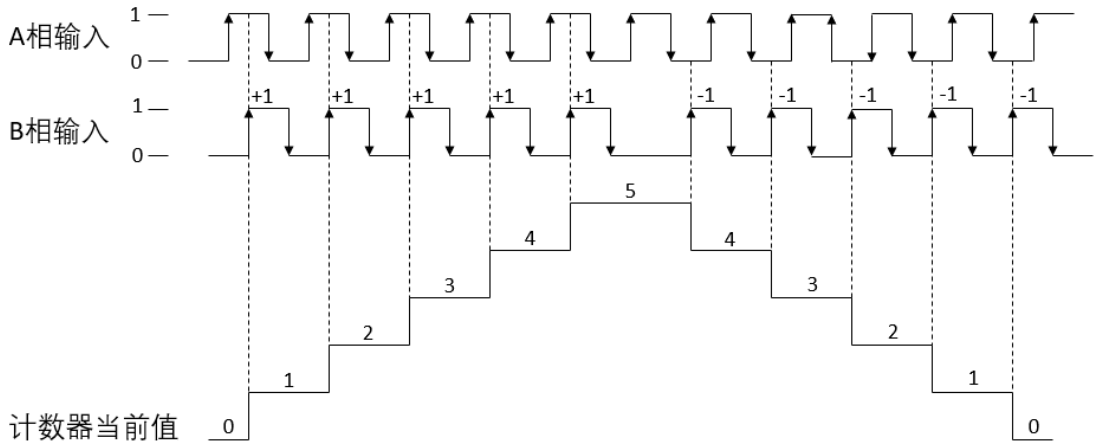


2)AB 相模式

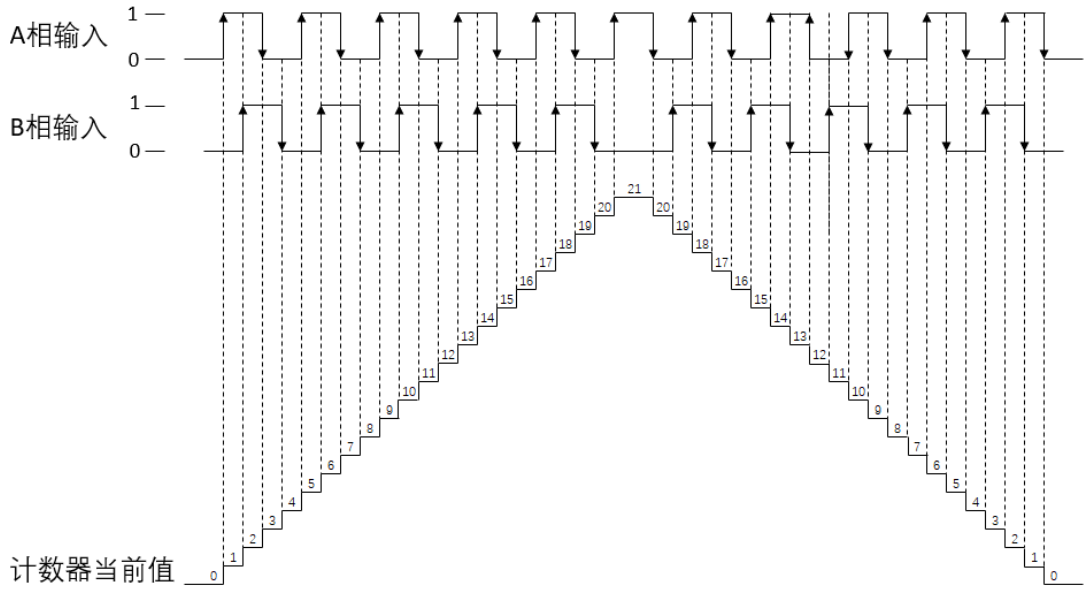
此模式下，高速计数值依照相位差90° 的脉冲信号（A相和B相）进行递增或递减计数，根据倍频数，又可分为一倍频和四倍频两种模式。

一倍频计数模式和四倍频计数模式分别如下：

- 一倍频模式



- 四倍频模式



6.4 高速计数模式选择

计数模式的选择由每个计数器对应的特殊寄存器（D8235~D8242）来指定，更改模式后，需要重新使能高速计数器才会生效。

高速计数器	C235	C236	C237	C238	C239	C240	C241	C242
特殊寄存器	D8235	D8236	D8237	D8238	D8239	D8240	D8241	D8242

模式与参数对应关系如下：

1：单相递增计数

2：单相递减计数

3：AB 相计数（1 倍频）

4：AB 相计数（4 倍频）

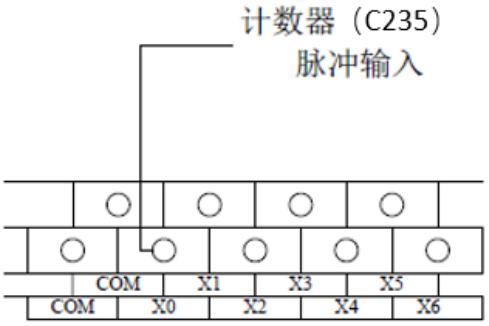
6.5 高速计数值范围

高速计数器计数范围为：K-2,147,483,648 ~ K+2,147,483,647。当计数值超出此范围时，则产生上溢或下溢。

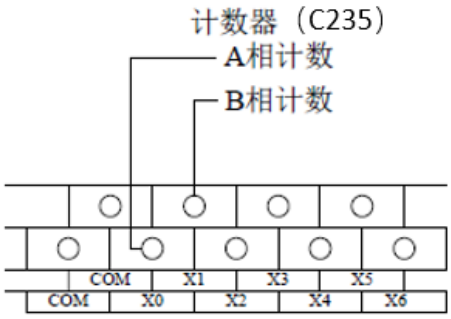
上溢：计数值从 K+2,147,483,647 跳转为 K-2,147,483,648，并继续计数。

下溢：计数值从 K-2,147,483,648 跳转为 K+2,147,483,647，并继续计数。

6.6 高速计数输入端接线



单相模式



AB相模式

6.7 高速计数相关指令

指令	说明	章节
HSCS	比较置位（高速计数器）	6.7.1 HSCS 指令(比较置位)
HSCR	比较复位（高速计数器）	6.7.2 HSCR 指令(比较复位)
HSZ	区间比较（高速计数器）	6.7.3 HSZ 指令(区间比较)

6.7.1 HSCS 指令(比较置位)

每次执行时，都将高速计数器的计数值和指定值做比较，如果两个值相等，立即置位比较输出的指令。

1、指令格式

16bit	32bit 13 步	指令格式

\	\	DHSCS	\	DHSCS S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	与高速计数器的当前值比较的数据，或是保存比较数据的字软元件编号	BIN32 位
S2	高速计数器的软元件编号[C235～C242]	BIN32 位
D	一致后进行置位(ON)的位软元件编号或中断号	位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2														
D		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2						●							●	
D													●	●

2、功能和动作说明

（1）当 S2 计数器的当前值等于设定值 S1 时，立即置位 D 或跳到中断子程序。S2 变量必须为高速计数器 C235~C242，因为涉及到的计数器均为 32 位，故必须采用 32 位指令 DHSCS。

（2）D 为普通位元件时：当为 Y0~Y17 范围端口时为立即输出；当为 Y20 以后的端口时，会等本次用户程序扫描完毕才会输出；当为 M、S、B、L、F、Dx.y 变量时，立即刷新。

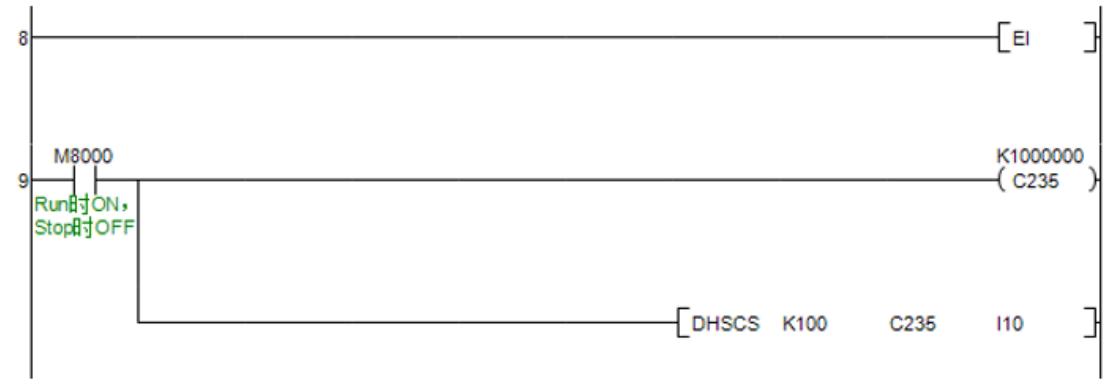
（3）D 为调用计数中断子程序指针时：当 D 为 I010~I050 时，即为调用高速计数器中断的子程序。当然必须写好相应的中断子程序、开启相应中断允许标志和全局中断允许标志等，才能正常相应计数器中断。M8059 置 ON 则禁止了所有的高速计数器中断（I010~I050）。

3、程序举例

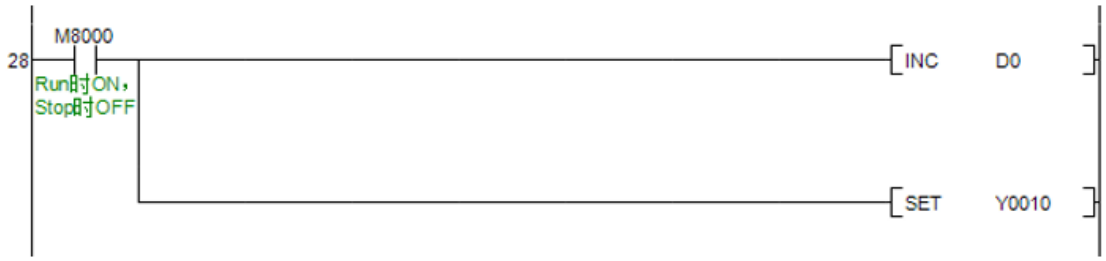
（1）置位普通位元件

（2）调用计数中断子程序

主程序



I010中断子程序



4、注意事项

（1）使用 HSCS 指令时，应保证所使用的计数器已被启用，否则计数器的值将不会变化；

（2）计数器是以中断方式响应计数器的输入信号，及时比较，若本次比较时满足匹配关系，比较输出立即置位。例如上面例一中，若 C235 的当前值变为 99->100 或 101->100 时，Y10

立即置位，且一直保持该状态，之后即使 C235 与 K100 的比较结果变成不相等，Y10 仍然保持 ON 状态，除非有另外的复位操作。

（3）指令的比较输出结果只决定脉冲于输入时的比较结果，即使采用 DMOV、DADD 等指令改写 C235~C242 的内容，若没有脉冲输入，比较输出也不会变化。

（4）当 HSCS 指令的输出目标为计数器中断 I010~I050 时，每个中断号只能使用 1 次，不可重复。

（5）HSCS、HSCR、HSZ 与普通指令一样可以多次使用，但这些指令同时驱动的个数限制在 6 个指令一下。

6.7.2 HSCR 指令(比较复位)

每次执行时，都将高速计数器的计数值和指定值做比较，如果两个值相等，立即复位比较输出的指令。

1、指令格式

16bit		32bit 13 步		指令格式
\	\	DHSCR	\	DHSCR S1 S2 D

操作数的数据类型如下表

操作数	内容	类型
S1	与高速计数器的当前值比较的数据，或是保存比较数据的字软元件编号	BIN32 位
S2	高速计数器的软元件编号[C235～C255]	BIN32 位
D	一致后进行复位(OFF)的位软元件编号	位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2														
D		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	

S2						●							●	
D													●	

2、功能和动作说明

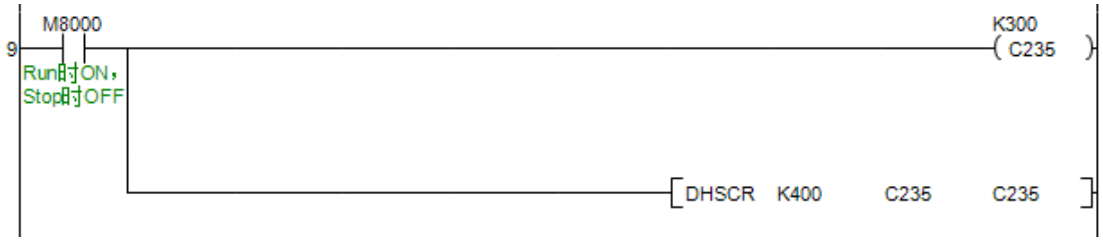
（1）当 S2 计数器的当前值等于设定值 S1 时，立即复位 D。S2 变量必须为高速计数器 C235~C242，因为涉及到的计数器均为 32 位，故必须采用 32 位指令 DHSCR。

（2）D 为普通位元件时：当为 Y0~Y17 范围端口时为立即输出；当为 Y20 以后的端口时，会等本次用户程序扫描完毕才会输出；当为 M、S、B、L、F、Dx.y 变量时，立即刷新。

（3）D 为计数器时：可以复位任意计数器，常用于复位本体高速计数器。

3、程序举例

自我复位的梯形图实例 C235 的当前值变为 400 后，立即执行 C235 的复位，当前值为 0，输出触点为 OFF。



4、注意事项

（1）使用 HSCSR 指令时，应保证所使用的计数器已被启用，否则计数器的值将不会变化；

（2）计数器是以中断方式响应计数器的输入信号，及时比较，若本次比较时满足匹配关系，比较输出立即复位。

（3）指令的比较输出结果只决定脉冲于输入时的比较结果，即使采用 DMOV、DADD 等指令改写 C235~C242 的内容，若没有脉冲输入，比较输出也不会变化。

（4）HSCS、HSCR、HSZ 与普通指令一样可以多次使用，但这些指令同时驱动的个数限制在 6 个指令一下。HSZ 指令特殊模式（高速表格比较模式、频率控制模式）仅能同时驱动 1 个指令。

6.7.3 HSZ 指令(区间比较)

将高速计数器的当前值和 2 个值(区间)进行比较，并将比较结果输出(刷新)位软元件(3 点)中。也可将该指令设置成高速表格比较模式和频率控制模式。

1、指令格式

16bit		32bit 17 步		指令格式
\	\	DHSCZ	\	DHSCZ S1 S2 S D

操作数的数据类型如下表

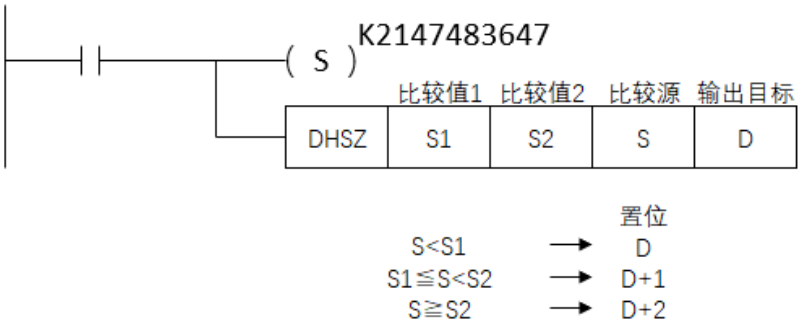
操作数	内容	类型
S1	比较区间下限	BIN32 位
S2	比较区间上限	BIN32 位
S	高速计数器的软元件编号[C235～C255]	BIN32 位
D	输出与比较上限值和比较下限值比较的结果的起始位软元件编号	位

操作数的对象软元件如下表

操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S1											●	●		
S2											●	●		
S														
D		●	●			●	●	●	●	●				
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S1	●	●	●	●	●	●	●	●	●	●	●	●	●	
S2	●	●	●	●	●	●	●	●	●	●	●	●	●	
S						●							●	
D													●	

2、功能和动作说明

（1）普通模式



• S1 为区间下限，S2 为区间上限，满足 S1<=S2；

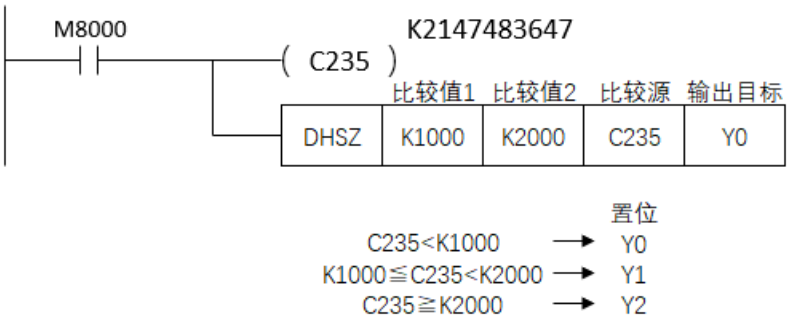
• S 必须为 C235~C242；

• D 占用三个连续地址单元，当为 Y0~Y17 端口范围时，立即输出，其他 Y 端口在本次用户程序扫描完毕才输出；当为 M、S、B、L、F、Dx.y 变量时，为立即刷新。

3、程序举例

（1）普通模式

高速计数器 C235 的当前值如下变化(计数)时，在 Y000、Y001、Y002 的任意一个中输出比较结果。



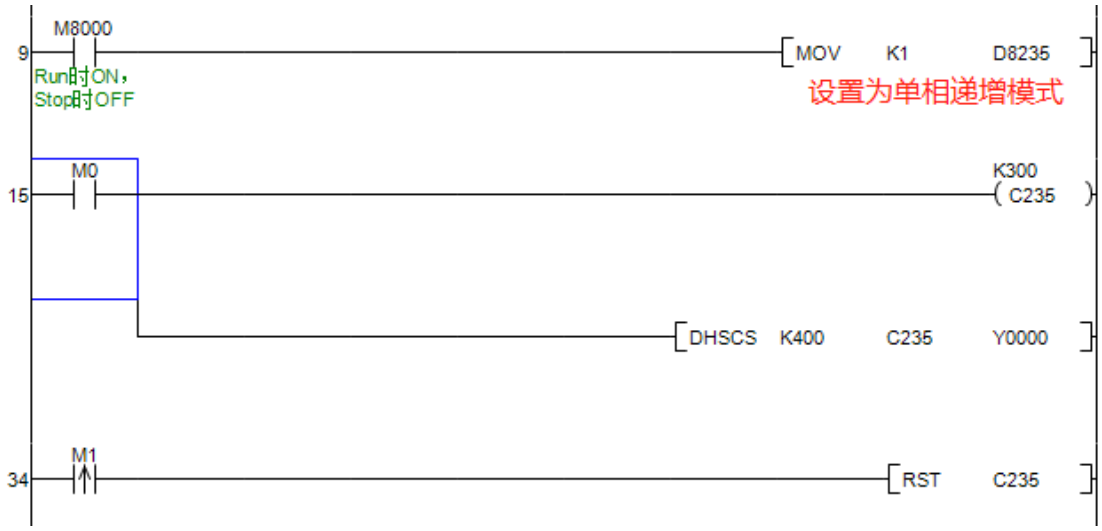
4、注意事项

（1）HSCS、HSCR、HSZ 与普通指令一样可以多次使用，但这些指令同时驱动的个数限制在 6 个指令一下。

（2）普通模式下，即使高速计数器没有打开也会进行区间比较置位对应软元件。

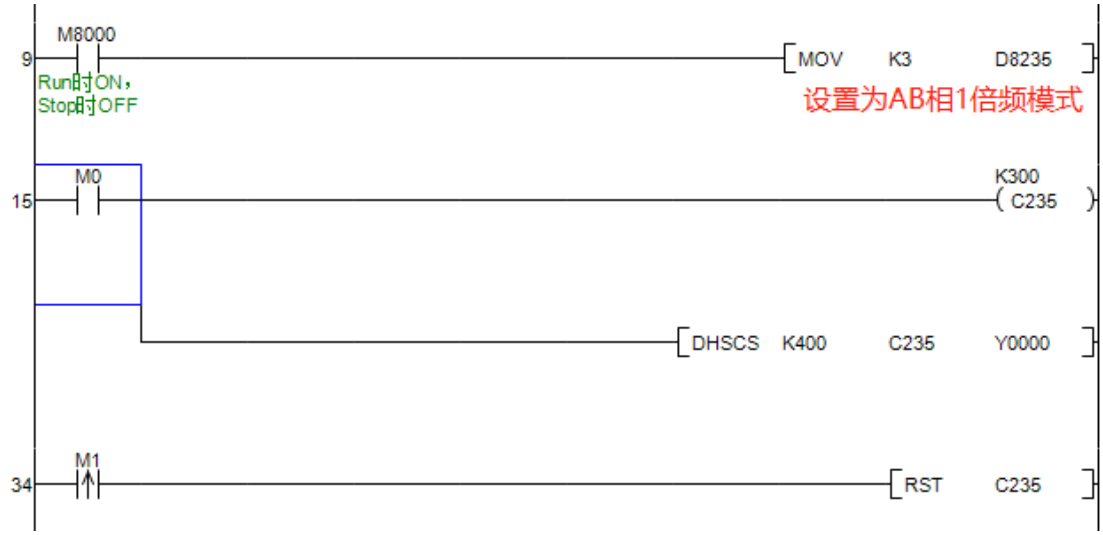
6.8 高速计数相关例程

1. 单相递增模式



- C235 在 M0 为 ON 时，对输入 X0 端口的 OFF→ON 上升沿进行高速计数，当计数值等于 K400 时，置位 Y0。
- 当 M1 状态 OFF→ON 上升沿时，高速计数器 C235 复位，同时 C235 寄存器值清零。
- 单相递减模式将上例中的 D8235 赋值 K2 即可。

2. AB 相模式



- C235 在 M0 为 ON 时，对输入 X0(A 相)、X1(B 相)端口的 OFF→ON 上升沿进行高速计数，当计数值等于 K400 时，置位 Y0。
- 当 M1 状态 OFF→ON 上升沿时，高速计数器 C235 复位，同时 C235 寄存器值清零。
- AB 相 4 倍频模式将上例中的 D8235 赋值 K4 即可。

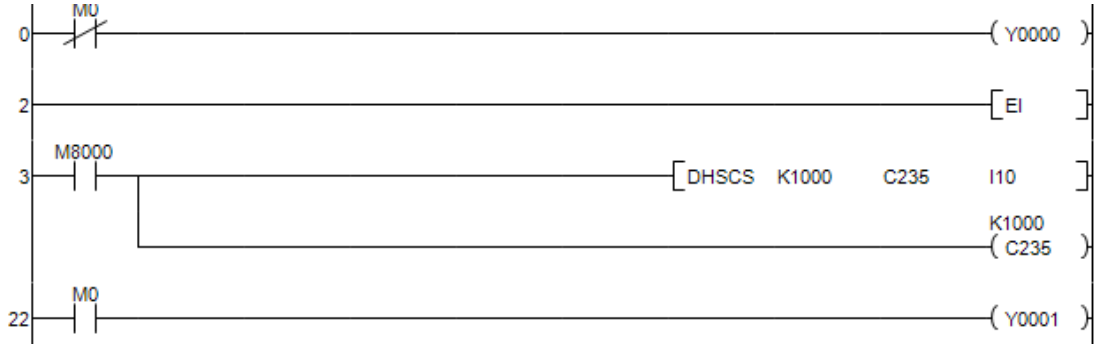
6.9 高速计数中断

根据高速计数器用比较置位指令(DHSCS 指令)的比较结果，执行中断子程序。用于使用高速计数器优先处理计数结果的控制。

指针编号	中断禁止标志位
I010	SM8059
I020	
I030	
I040	
I050	

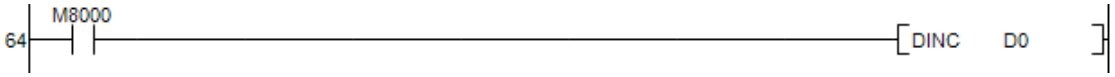
程序举例：

主程序



当 C235 的值等于 K1000 时，立即执行中断程序 I010。

I010 中断计数子程序



7 通讯

7.1 MODBUS 主站

1、指令格式

16bit 6 步		32bit 6 步		指令格式
MODBUS	\	\	\	MODBUS S D

操作数的数据类型如下表

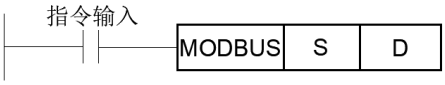
操作数	内容	类型
S	端口编号，K0-K15 表示端口编号，K300 表示 TCP	实数(2 进制)
D	行号，通信数据都以表格方式在主站设定界面设定	实数(2 进制)

操作数的对象软元件如下表

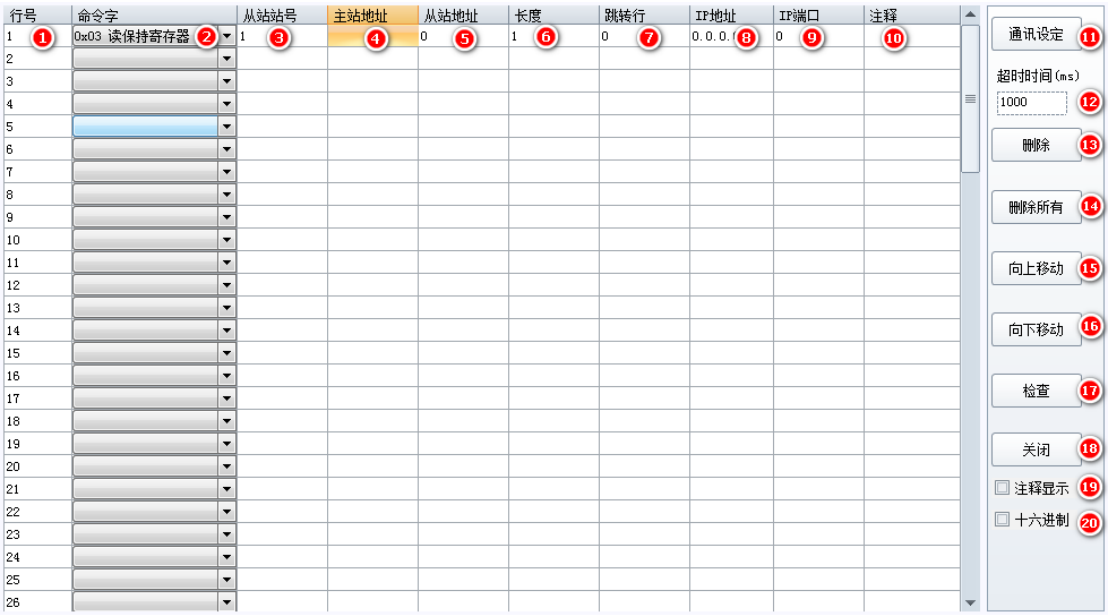
操作数种类	位软元件										常数		实数	字符串
	X	Y	M	T	C	S	B	L	F	Dx.y	K	H	E	“”
S											●	●		
D											●	●		
操作数种类	字软元件										变址			指针
	KnX	KnY	KnM	KnS	T	C	D	R	VD	RD	V	Z	修饰	P
S							●	●					●	
D							●	●					●	

2、功能和动作说明

MODBUS 主站指令



主站表格参数设置



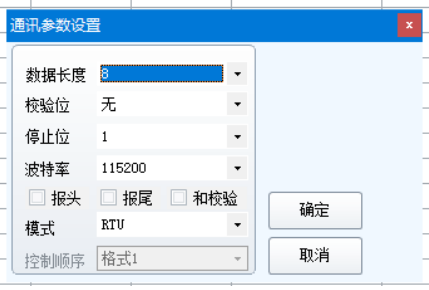
1	行号（1~99），作为 MODBUS 指令的参数索引
2	命令字 0x01 读线圈 0x02 读离散量输入 0x03 读保持寄存器 0x04 读输入寄存器 0x05 写单个线圈 0x06 写单个寄存器 0x0F 写多个线圈 0x10 写多个寄存器
3	从站的站号（从 1 开始）
4	用于存放从从站接收的数据，或者存放要发送给从站的数据。 对于字命令可使用 D 或 R；对于位命令，可使用 M 位元件。
5	从站地址，可参照从站的寄存器定义设置
6	读写寄存器的长度
7	默认为 0，不可修改
8	服务器 IP 地址，Modbus/TCP 客户端用
9	服务器 IP 端口，Modbus/TCP 客户端用
10	注释说明

11	设置串口波特率校验位等参数
12	从站回复超时时间
13	删除当前行配置
14	删除所有配置
15	当前行配置向上移动
16	当前行配置向下移动
17	检查所有配置
18	关闭当前设置页面
19	勾选后，显示[4]主站地址注释
20	勾选后，[5]从站地址用 16 进制显示

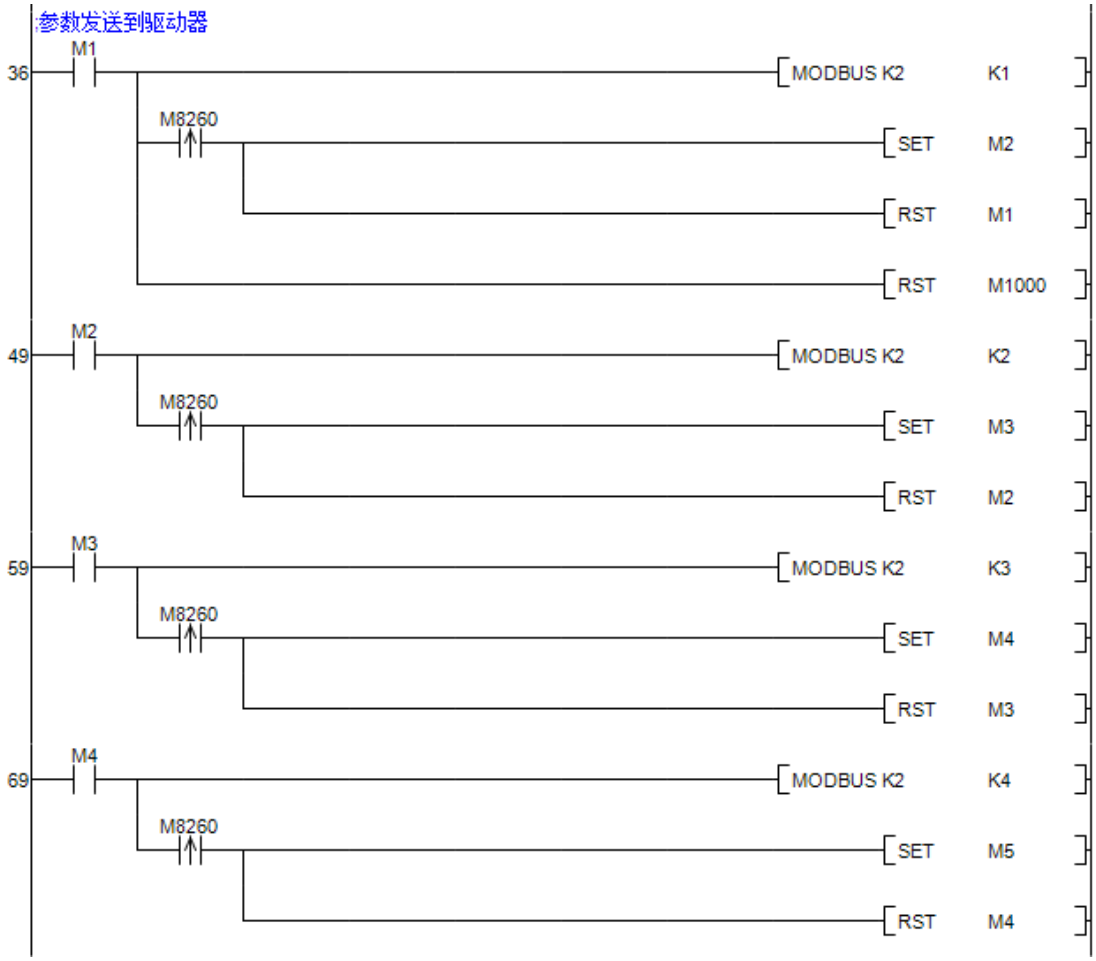
3、程序举例

MODBUS 主站表格配置

行号	命令字	从站站号	主站地址	从站地址	长度	跳转行	IP地址	IP端口	注释
1	0x06 写单个寄存器	2	D0	25088	1	0	0.0.0.0	0	运行模式
2	0x06 写单个寄存器	2	D1	25090	1	0	0.0.0.0	0	位置低16位
3	0x06 写单个寄存器	2	D2	25089	1	0	0.0.0.0	0	位置高16位
4	0x06 写单个寄存器	2	D3	25091	1	0	0.0.0.0	0	运行速度
5	0x06 写单个寄存器	2	D4	25092	1	0	0.0.0.0	0	加速时间
6	0x06 写单个寄存器	2	D5	25093	1	0	0.0.0.0	0	减速时间
7	0x06 写单个寄存器	2	D6	25094	1	0	0.0.0.0	0	停顿时间
8	0x06 写单个寄存器	2	D10	24578	1	0	0.0.0.0	0	控制字16
9	0x06 写单个寄存器	3	D0	25096					
10	0x06 写单个寄存器	3	D1	25098					
11	0x06 写单个寄存器	3	D2	25097					
12	0x06 写单个寄存器	3	D3	25099					
13	0x06 写单个寄存器	3	D4	25100					
14	0x06 写单个寄存器	3	D5	25101					
15	0x06 写单个寄存器	3	D6	25102					
16	0x06 写单个寄存器	3	D11	24578					17
17	0x06 写单个寄存器	4	D0	25104					
18	0x06 写单个寄存器	4	D1	25106					
19	0x06 写单个寄存器	4	D2	25105					



程序



4、注意事项

无

7.2 字符集

7.2.1 OPENS 指令(打开端口)

1、指令格式

16bit 6 步		32bit 6 步		指令格式
OPENS	\	\	\	OPENS “S”

操作数的数据类型如下表

操作数	内容	类型
S	字符集表格内的“配置名称”	

7.2.2 SENDS 指令(发送)

1、指令格式

16bit 6 步		32bit 6 步		指令格式
SENDS	\	\	\	SENDS “S”

操作数的数据类型如下表

操作数	内容	类型
S	字符集表格内的“配置名称”	

7.2.3 REVS 指令(接收)

1、指令格式

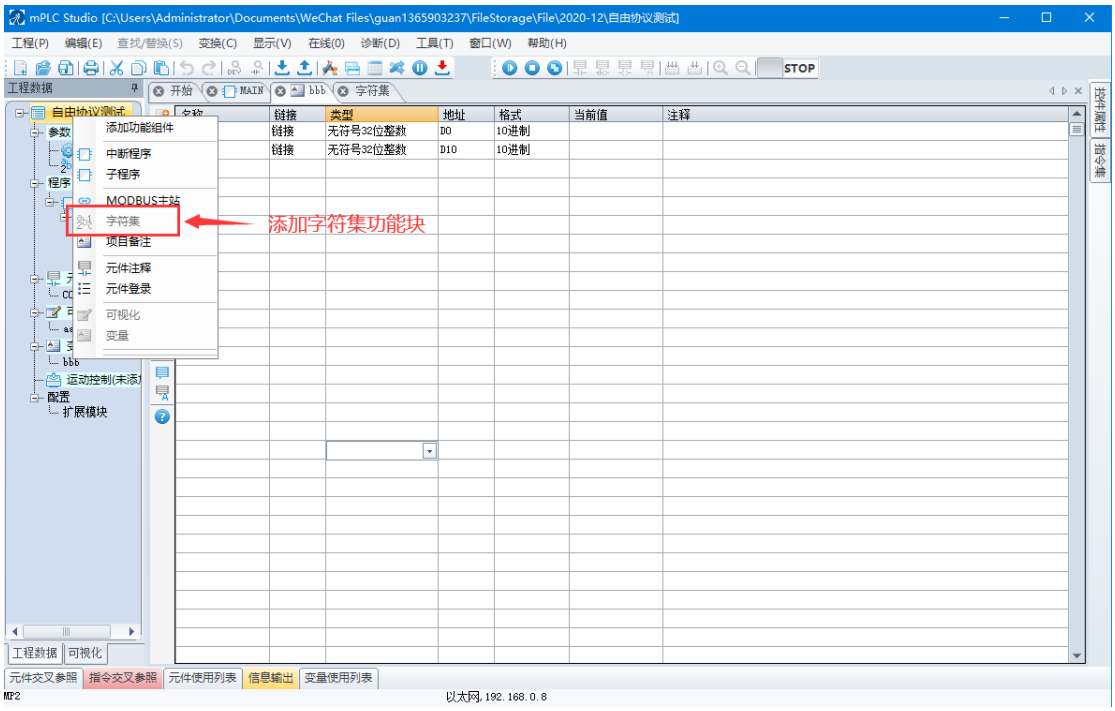
16bit 6 步		32bit 6 步		指令格式
REVS	\	\	\	REVS “S”

操作数的数据类型如下表

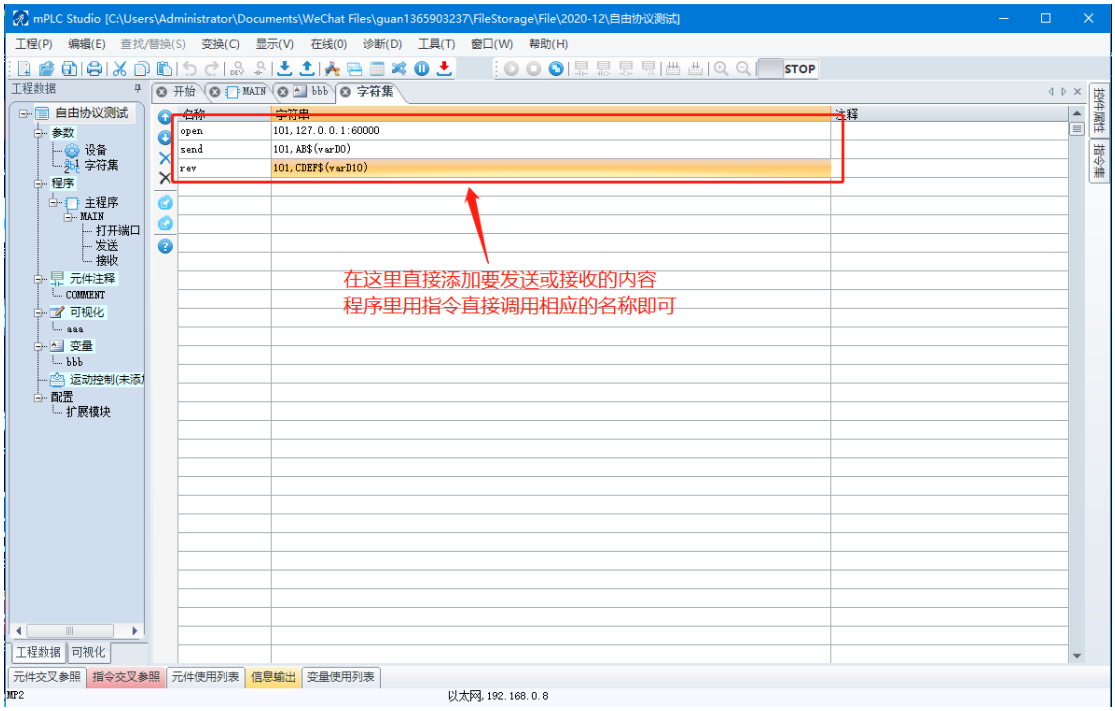
操作数	内容	类型
S	字符集表格内的“配置名称”	

7.2.4 字符集使用说明

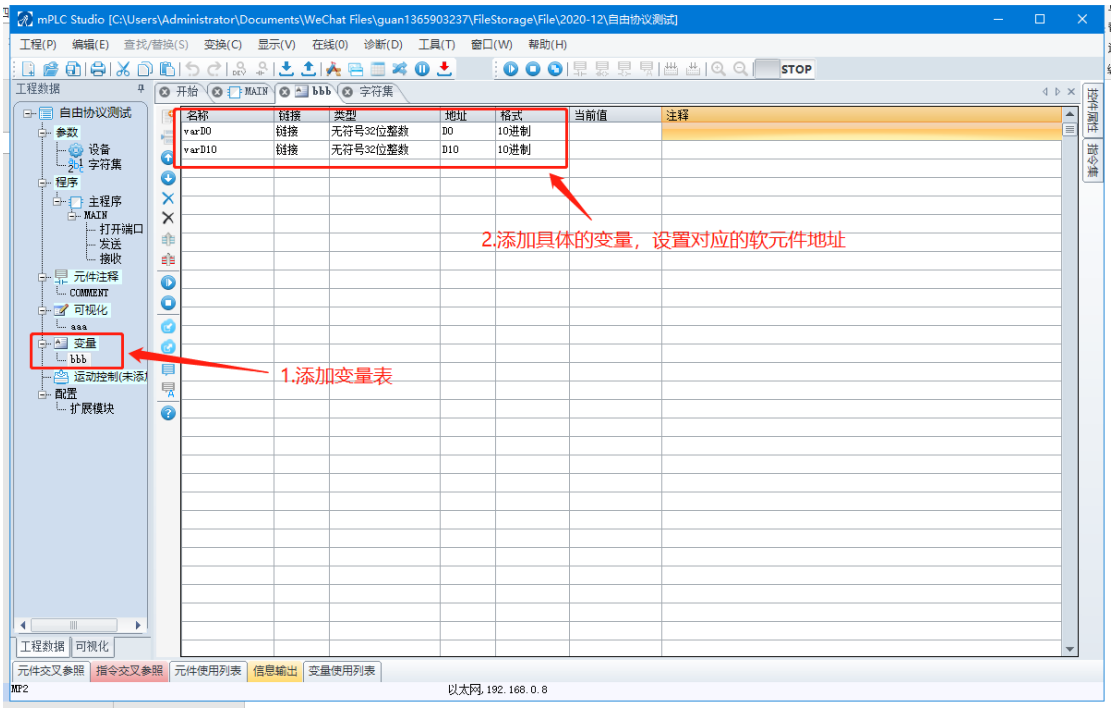
1.编写字符集程序。新建项目工程程序，在左侧项目数据中添加字符集功能。



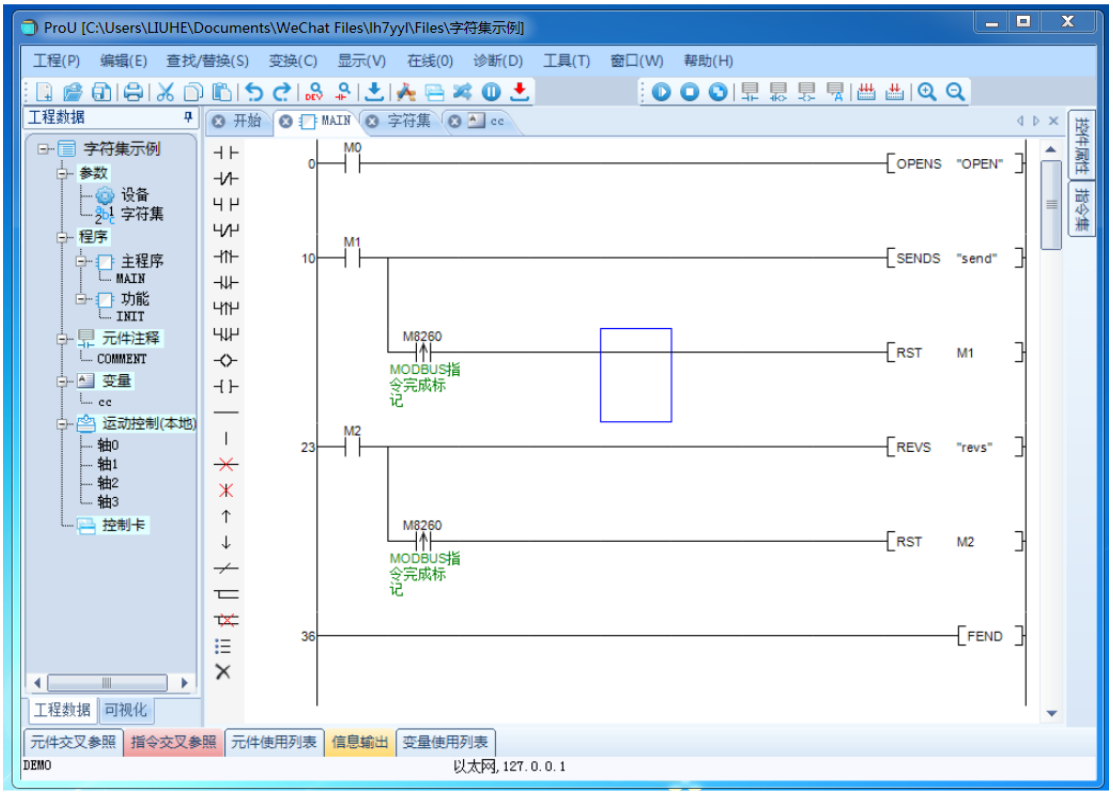
2. 打开字符集设置界面，字符集的数据发送和接收均以填表方式编写。程序中调用相应的行即可。打开端口按固定格式填写：100 表示以太网识别号（100~109），127.0.0.1 表示对应 IP 地址；8000 表示对方端口号； send 行表示：字符串 ABC 和变量 varD0 中的内容通过 100 端口发送； revs 行表示：通过端口 100 接收字符串 CDEF 和数据存放到变量 varD10 中。



3. 设置变量，varD0 映射到 D0 软元件，varD10 映射到 D10 软元件



4. 编写梯形图程序。 OPENS 指令表示打开端口，需要一直保持接通状态；SENDS 指令表示将 send 字符集中的内容发送到端口，发送成功后可用 M8029 标志位进行复位；REVS 指令表示从端口接收到的内容，接收成功后可用 M8029 标志位进行复位。程序编写完成后，需要下载到 PLC，并重启。



8 附录

8.1 错误码内容

错误类型	错误时动作	错误代码	代码解释
PLC 硬件错误	停止运行	6130	硬件不支持设定的串口通信波特率
		6131	配置外扩 RAM 失败
		6132	FLASH 配置失败
		6133	网络初始化失败
		6134	写入数据到 Flash 错误
		6135	扩展 IO 模块断开
		6136	串口初始化失败
		6354	寄存器操作超出支持的范围
		6355	脉冲输出忙

		6356	串口通信器操作超出支持的端口范围
		6357	设置时间的参数格式不正确
		6358	串口通信数据校验和错误
		6359	网络通信数据错误
		6360	输出端口操作范围超出硬件支持的最大范围
		6361	PWM 输出频率超出硬件支持范围
		6362	PWM 输出轴号超出硬件支持范围
		6363	执行运动指令失败
		6364	串口接收数据溢出
		6365	轴的编号超出硬件支持的最大范围
		6366	轴偏移数据缓冲区满
		6367	轴正在回零中，设置轴位置失败
		6368	轴位置设置失败
		6369	获取轴位置失败
		6370	网络接收数据错
		6371	网络接收缓冲区溢出
		6372	正在进行写操作
		6373	扩展模块收到错误的数据包
		6374	扩展模块数据包丢失
		6375	不合法的数据包
		6376	运行停止失败
		6377	保留
		6378	不存在的扩展 IO 输入模块
		6379	不存在的扩展 IO 输出模块
		6380	系统断电保存数据丢失 ， 解决方法：下载程序，错误自动清除

		6381	扩展 IO 模块断开 ， 解决方法：查看网络是联通，重新启动主机，进行扩展模块扫描
		6382	扩展模块配置数据下载错误，解决方法：读 PLC 配置，然后下载配置数据，错误自动清除
		6383	扩展模块类型不一致，解决方法：读 PLC 配置，然后下载配置数据，错误自动清除
		6384	扩展模块数量不一致，解决方法：读 PLC 配置，然后下载配置数据，错误自动清除
		6385	系统时钟丢失 ，解决方法： 进入编程软件，在线 -》 设置时钟 -》读取时钟 -》设置，错误自动清除
		6386	保留
		6387	保留
		6388	保留
		6389	保留
		6390	保留
		6391	保留
		9392	保留
		6393	保留
		6394	保留
		6395	保留
		6396	保留
		6397	保留
		6398	保留
		6399	保留
		6400	保留